Hierarchical Image Descriptions for Classification and Painting

Song, Yi-Zhe

Award date:
2009

Awarding institution:
University of Bath

Hierarchical Image Descriptions for Classification and Painting

submitted by

Yi-Zhe Song

for the degree of Doctor of Philosophy

of the

University of Bath

March 2009

Hierarchical Image Descriptions for Classification and Painting

Yi-Zhe Song

SUMMARY

The overall argument this thesis makes is that *topological object structures captured within hierarchical image descriptions are invariant to depictive styles and offer a level of abstraction found in many modern abstract artworks.*

To show how object structures can be extracted from images, two hierarchical image descriptions are proposed. The first of these is inspired by perceptual organisation; whereas, the second is based on agglomerative clustering of image primitives. This thesis argues the benefits and drawbacks of each image description and empirically show why the second is more suitable in capturing object strucutures. The value of graph theory is demonstrated in extracting object structures, especially from the second type of image description. User interaction during the structure extraction process is also made possible via an image hierarchy editor.

Two applications of object structures are studied in depth. On the computer vision side, the problem of object classification is investigated. In particular, this thesis shows that it is possible to classify objects regardless of their depictive styles. This classification problem is approached using a graph theoretic paradigm; by encoding object structures as feature vectors of fixed lengths, object classification can then be treated as a clustering problem in structural feature space and that actual clustering can be done using conventional machine learning techniques.

The benefits of object structures in computer graphics are demonstrated from a Non-Photorealistic Rendering (NPR) point of view. In particular, it is shown that topological object structures deliver an appropriate degree of abstraction that often appears in well-known abstract artworks. Moreover, the value of shape simplification is demonstrated in the process of making abstract art. By integrating object structures and simple geometric shapes, it is shown that artworks produced in child-like paintings and from artists such as Wassily Kandinsky, Joan Mirö and Henri Matisse can be synthesised and by doing so, the current gamut of NPR styles is extended. The whole process of making abstract art is built into a single piece of software with intuitive GUI.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincerest gratitude to my supervisor, Dr. Peter Hall, for all of his guidance, patience, encouragement and friendship. I wish to thank him for showing me the door to research almost 6 years ago when he was my undergraduate project supervisor and for such a lovely ride during my PhD studies. It would not have been possible without him.

Many thanks are also due to my friends and colleagues: John Collomosse, for his insightful advices; Lin Li, for simply being there for me during the last 10 years; Chuan Li and Dan Rao, for giving me a place in their life and dinner table; Peiyi Shen and Qiang Du, for their support and encouragement; Fynn Scheben, Anupriya Balikai and Dan Beale for proofreading my thesis.

Last, but not least, I want to thank my parents for raising me up, teaching me to be kind, helpful to others and providing me with the best they could offer and beyond. Thanks are also due to my wife, Jingge Hu, for her love and support along the way.

I gratefully acknowledge the financial support of the EPSRC who funded this research under grant "Matching and Painting using Gestalt models" (EP/D05429X/1).

PUBLICATIONS

- Y.-Z. Song and P. M. Hall. Stable Image Descriptions using Gestalt Principles. In G. Bebis et al. (Eds.): International Symposium on Visual Computing (ISVC) 2008, Part I, LNCS 5358, pp. 318-327, 2008. Las Vegas, Nevada, USA.

- X. Bai, Y.-Z. Song, A. Balikai, and P. M. Hall. Structure is a Visual Class Invariant. In N. da Vitora Lobo et al. (Eds.): Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition (SSPR&SPR) 2008, LNCS 5342, pp. 329-338, 2008. Orlando, Florida, USA.

- A. Balikai, P. L. Rosin, Y.-Z. Song and P. M. Hall. Shapes Fit For Purpose. British Machine Vision Conf., Leeds, UK, pp. 443-452, 2008.

- Y.-Z. Song, P. L. Rosin, P. M. Hall, and J. P. Collomosse (2008). Arty shapes. In P. Brown, D. W. Cunningham, V. Interrante, and J. McCormack (Eds.), Computational Aesthetics in Graphics, Visualization, and Imaging, Lisbon, Portugal, pp. 65–72. Eurographics Association.

- X. Bai, Y.-Z. Song, and P. M. Hall. Learning object classes from structure. In British Machine Vision Conference 2007, Warwick, UK, pp. 322–330.

- P. M. Hall, J. P. Collomosse, Y.-Z. Song, P. Shen, and C. Li (2007). Rtcams: A new perspective on nonphotorealistic rendering from photographs. IEEE Transactions on Visualization and Computer Graphics 13 (5), 966–979.

- Y.-Z. Song and C. P. Town. Visual recognition of man-made materials and structures in an office environment. In P. Hall and P. Willis (Eds.), Vision, Video, and Graphics, Edinburgh, UK, pp. 32–40. Eurographics Association, 2005.

Contents

Part I

Introduction

Chapter 1

Introduction

1.1 Objectives

The ultimate goal of this work is to study the profound relationship amongst photographs, paintings and drawings, so that we can easily travel across the semantic boundaries in-between. In particular, the aim of this thesis is three-fold: (i) identify the invariant property among objects depicted in such different styles; (ii) propose ways of automatically extracting this property from images; (iii) and finally use it in novel applications of object classification across depictions and synthesising abstract artworks from photographs. In this thesis, it is proposed that topological object structure captured within hierarchical image descriptions is a key property that is shared amongst objects of different depictive styles.

On one side, success in classifying objects regardless of depiction would greatly benefit computer vision, where photographs have been the dominant research subject; on the other side, the Non-Photorealistic Rendering (NPR) literature will also benefit from the high-level abstraction that object structures carry, so that arts of an abstract nature can be synthesised from photographs.

1.2 Motivations

In ancient times, people used to express themselves by drawing on the surface of caves. An example of such cave painting is exhibited in Figure 1-1. With

Figure 1-1: An example cave drawing: "Hunting"

their limited tools, such type of drawings often lack in their visual richness: they are usually in only one colour and objects are often lack of details. Despite all that, they still carry a rich amount of visual information. Even now, after thousands of years, we are still able to tell the same story that was intended in such cave paintings; we are still able to appreciate the bravery of ancient hunters expressed in Figure 1-1 and imagine the kind of life style they were living by. Why? Because such paintings contain an appropriate amount of abstraction that is still perceivable to us nowadays.

Master artists are extremely good in delivering abstraction in their works as well. Figure 1-2 shows how Picasso depicts a bull and how Joan Mirö paints a human figure. The artwork themselves are highly abstract, however, objects in them can still be correctly perceived. Why? Again, the amount of abstraction they introduced in their paintings is sufficient for us to perceive and will certainly remain so for thousands of years to come.

In fact, non-artists, like the author himself and many others, tend to abstract well too. A typical task would be to draw a stickman. Figure 1-3 offers two of them: one drawn by the author using GIMP and the other by a 3-year-old girl using a red-ink pen.

However, unlike cave paintings, abstract artworks and the author's stickman, photographs offer no abstraction at all, but an highly detailed representation of a particular visual scene. Probably the easiest thing to do nowadays would be to take a digital photograph, which often only involves pressing down the shutter button. The author made use of a digital camera and it took him less than 1 minutes from taking the photo in Figure 1-4 to displaying it on a PC. With the

Figure 1-2: Left: Picasso's version of a bull; right: an example of how Joan Mirö paints a human figure.

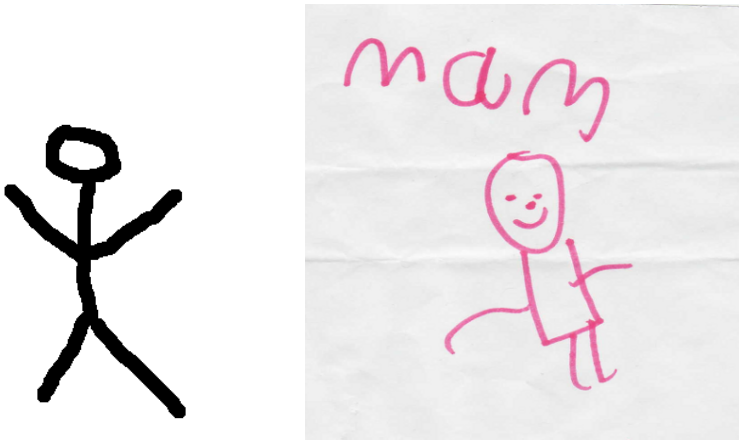


Figure 1-3: Two stickman figures draw by the author and a 3-year-old girl

vastly increasing availability of digital photographic equipments such as digital cameras and camcorders, digital photographs have become ever so popular.

Nevertheless, we as humans are extremely powerful when it comes to interpreting visual information. Indeed, no matter how a human figure might be depicted, whether it is in an ancient cave drawing (Figure 1-1), an abstract painting (Figure 1-2), a few stickmen (Figure 1-3) or a photograph (Figure 1-4), we are still able to perceive them correctly. This automatically leads to this question "what is the invariant property that is captured?" We vaguely observed that abstraction could be the key in this context. However, the key question now comes to "how do we abstract appropriately, so that objects still have their semantic meaning intact?"

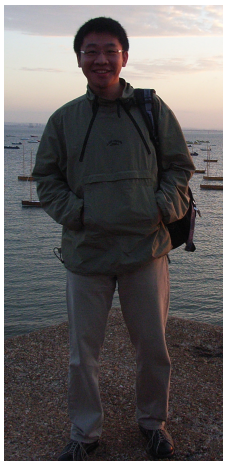


Figure 1-4: A photograph of the author

As a start, we shall seek an answer to the above questions by analysing how master artists such as Joan Mirö introduce abstraction. The following quote [36] is from Joan Mirö himself on how he approaches creating abstract artworks:

> "When I paint, I try to determine a relationship between the strokes and the colours, a balance that gives life and that lives in terms of that relationship. ... All the things in the painting are organised to meet the demand for balance, and every part is linked in a precise relationship. ... I like to speak in lines - strokes - and colours. A line, a colour - a woman; a line, a colour - a bird. It is the drawing and the painting that, by means of a symbol, suggest the idea of the woman and the idea of the bird."

In the above quote, the master artist talked about how he balances strokes to create meaning; and more importantly, how he uses symbols to represent objects. In addition to those, there is a lot more to his works, some of which we are only in the position to appreciate, rather than to understand or to mimic. Nevertheless, Mirö mentioned at many places the importance of "balance", "relationship" of parts and how he produces his abstract paintings by organising simple elements, such as lines, shapes, strokes, in a structured fashion to create life. In fact, from observing many symbolic abstract paintings from Mirö, we see that no matter how abstract his objects get, they will almost always follow the toplogical structures of objects. Therefore, it seems to be that "balance" and "relationship" of parts that the master is referring to partially derives from the actual topological structure

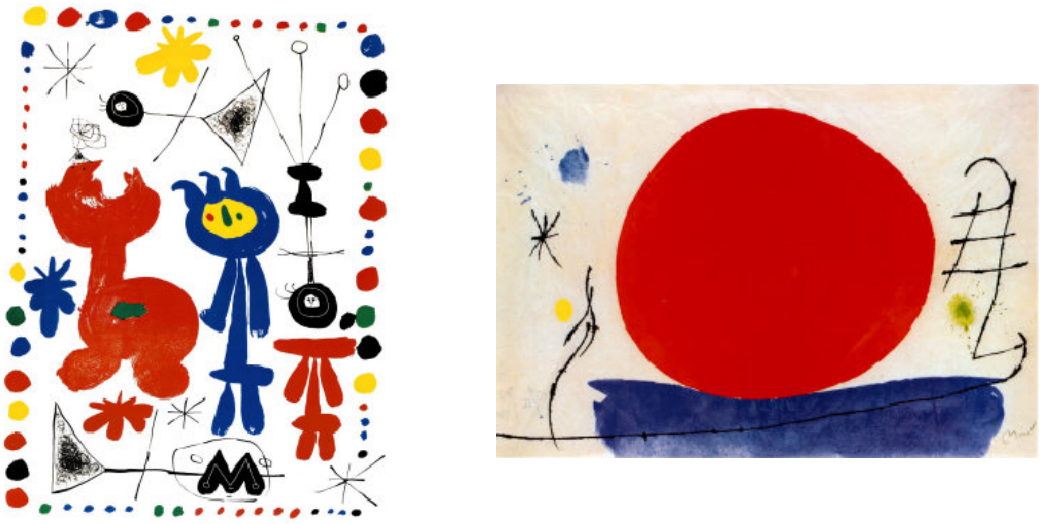


Figure 1-5: Two examples of Mirö's paintings. Left: "Personnage Et Oiseaux"; right: "Senza Titolo"

of objects. In other words, Mirö finds such topological structures an excellent tool in delivering a level of abstraction when depicting objects. A bird would normally have a body and two wings and a human figure would have a head, a body, arms, legs and so on; similarly a boat would have a bottom, a front side and a back side. Evidence of such can be found in paintings such as "Personnage Et Oiseaux" and "Senza Titolo", both exhibited in Figures 1-5.

We can also consider a thought experiment using four images. Three images show faces, one little more than a scribble by a young child, one the face of a clown with crosses for eyes, the other a photograph. The fourth image is a photograph of a car. Which is the odd one out? We have run this experiment in practice, and were not at all surprised to find the car was always selected as the odd one out. Yet the variance between the faces is profound. Eyes can be any shape, so can the mouth. The drawn face may or may not have an outline, and if it does exist can be any shape. Children tend to draw the eyes at the top of the head when in fact they are in the middle. Given so much variation in shape, in position, in colour, it is difficult to see any invariant except the structural arrangement of facial parts.

So we as human beings are excellent at interpreting abstract visual information, we are able to categorise objects in Figures 1-1, 1-2, 1-3, 1-4 as human figures, but how well can the computers do in this case? It will be very interesting if there is a computer program that is able to categorise beyond object depiction

styles, so that a face is always a face no matter depicted in a drawing or painting or photograph. Given a photograph, it would also be interesting if we can automatically synthesise a drawing or painting from it, especially those of an abstract nature. This thesis makes an attempt to answer these two questions, which are essentially two sides of the same coin. Successfully addressing the first question will partially make way for other applications such as extending the capability of contemporary content-based image retrieval systems, which are largely limited to photographs; while answering the second question will extend the current gamuts of non-photorealistic rendering styles, towards the direction of producing more abstract art. We believe that topological object structures is the key in linking objects that were depicted differently.

1.3 Challenges

There are several challenges which need to be addressed in all three areas, viz, automatically extracting object structures, object classification across different depictive styles and synthesising abstract artworks:

- Objects can be depicted rather differently; as shown in Figures 1-1, 1-2, 1-3, 1-4, a human can exist in various depictions.
- It is widely known that the problem of segmenting objects from photographs alone still remains challenging and unresolved. In order to bridge the gap between photographs and paintings or drawings, the proposed algorithm would have to work relatively well on objects of various depictive styles, not only photographs.
- Object parts might be occluded in any depictive mode, sometimes even deliberately in the spirit of making abstract art. This results in the incompleteness of topological structures, which causes potential problems when we want to classify for example.
- The object structure extraction process needs to be as automatic as possible and should not require a lot of parameter tuning.
- Even a decent automatic object structure extraction algorithm would entail failure modes, which create ambiguities when the automatically created

structures are consequently used.

- Making art, especially making abstract art, is a challenging task on it own right. We need an easy and intuitive way to facilitate this process.

In this thesis, we address some of these difficulties, but not all. A summary of our contributions are provided next.

1.4 Contributions

The contributions of this thesis can be divided into three main categories, which are summarised below. A more detailed version is available in Section 8.1 towards the end of this thesis.

Extraction of object structures: To solve the problem of automatically extracting object structures, we proposed two hierarchical image descriptions in turn. The first of which is based on the psychological theory of perceptual organisation, whereas the second image description is obtained using agglomerative clustering. Both image descriptions are quantitatively evaluated using a novel experimental setup that uses human disagreement as a unit measure. The benefits and drawbacks of both techniques are then discussed in depth, which leads to a default choice that is used in later applications. We also offer an editing framework to accommodate any errors in the automatically extracted structures.

Categorising objects of different depictive styles: In object classification, we treat each automatically extracted object structure as a graph of nodes and arcs. We then proposed a graph theoretic technique that encodes each such graph into a vector of fixed length. Those feature vectors can then be clustered in feature space to form classes of objects. By doing so, we are able to accommodate partial differences in the automatically generated structures. We also offer some initial evidence on how representing objects as a collection of simple geometric shapes is able to aid in matching photographs to paintings and drawings.

Synthesising abstract artworks from images: We demonstrate that topological object structures are able to deliver a level of abstraction that is

often found in child drawings and abstract artworks from master artists such as Mirö, Picasso and Matisse. We build abstract representations of objects that combine object structures and simple geometric shapes. We then show that by rendering such representations appropriately, different styles of abstract artworks can be synthesised. A single piece of software with an intuitive GUI is implemented to facilitate the art creation process.

1.5 Structure of the Thesis

The thesis is divided into four main parts. In Part I, a general introduction of the thesis is given (Chapter 1), plus an in-depth literature review of the relevant research fields (Chapter 2). Part II (Chapter 3, 4, 5) of the thesis concentrates on introducing two hierarchical image descriptions which form the basis of extracting object structures and shows how manual interaction can be introduced to assist the extraction process. In Part III, two novel applications of the second hierarchical image description (Chapter 4) are investigated. The first of which uses object structures extracted from the image hierarchies to cluster objects depicted in different styles; the second application brings out the aesthetics side of object structures by synthesising arts of an abstract nature from photographs. The thesis is then concluded in Part IV, whereby insights on future work are offered as well.

We now provide an outline of the thesis chapter by chapter, summarising the main contributions in each and how they contribute to the overall argument that this thesis makes, which is "object structures captured within hierarchical image descriptions are invariant to depictive styles and offer a level of abstraction found in many modern abstract artworks".

Part I — Introduction
Chapter 1 — Introduction

In which we state the overall contribution of the thesis, that is we demonstrate that object structures are invariant to depictive styles and useful in generating synthetic abstract artworks. The motivation behind our contribution is discussed and a detailed organisation of the thesis is found at the end of the chapter.

Chapter 2 — Literature Review

We offer a detailed review of the field of research. Observations are made accordingly, so to identify gaps in the literature. The literature review will be split into two parts: from a computer vision perspective, it is argued that state-of-the-art image descriptions fall short when representing images of different depictive styles; from a computer graphics perspective, we will show how the gamut of artistic styles are limited in the current NPR literature, especially on approaching abstract artworks.

Part II — Hierarchical Image Descriptions and Object Structures

Chapter 3 — Stable Image Descriptions using Gestalt Principles

In which we introduce our first hierarchical image description, which is based on grouping image primitives. Its principal contribution is a general framework by which salient groups can be identified. Our approach begins not with pixels but with line segments and image regions of coherent colour. The grouping process is influenced by Gestalt principle, making use of proximity, common region and Prägnanz which was mathematically defined for the first time. We introduce the notion of "grouping scale" within a grouping hierarchy, and provide a measure for the salience of a group within that hierarchy. The salient groups selected by our method require just a single user parameter — which is the number of groups to be output. We empirically compare our approach with another based on normalised cuts which requires exactly the same user information, and also to human groupings. We demonstrate our groupings are closer to those from humans.

Chapter 4 — Hierarchical Agglomerative Clustering

We describe the second hierarchical image description this thesis offers. This particular image description is based on pair-wise merging of image primitives. Each step in the merging process offers a *rough* segmentation of visual objects. Additionally, we demonstrate the value of using a de-correlated feature vector when merging. Meaningful partitions can be found using a single parameter in a manner similar to that used in the first image description, i.e., the number of

partitions. In particular, we explain a graph theoretic approach that can be used to automatically stop the merging process, hence produce an image segmentation. This particular image description is also tested using the same novel experimental setup previously used in Chapter 3 and is shown to exceed the performance of the previous approach.

Chapter 5 — Interactive Editing

In which we propose ways of editing the automatically generated hierarchical image descriptions, especially of the type proposed in Chapter 4. Benefiting from the rich underlying image descriptions, the editing process is often easy and intuitive. A desired topological object structure can be obtained using a few mouse clicks.

Part III — Classification and Painting using Hierarchical Image Descriptions

Chapter 6 — Image Description for Object Classification: Structure is Invariant to Depictive Style

We show the value of hierarchical image descriptions in terms of the traditional computer vision task of object classification. Specifically, we demonstrate how structures extracted from hierarchical image descriptions enable us to cluster objects that are depicted in different styles. We depend on spectral graph analysis of an automatically extracted structure to construct a feature vector of fixed dimension, which can be classified using standard methods. Furthermore, we also demonstrate the value of representing objects and their parts by means of fitting simple geometric shapes and show that using such representation objects of different depictive styles can be successfully matched.

Chapter 7 — Image Description for Image Synthesis: Generating Abstract Artworks

In which we move on to demonstrate the value of topological object structures from a computer graphics perspective. Specifically we show how object structures are used to generate arts of an abstract nature. In addition, we will show the

value of optimal shape fitting in delivering abstraction to synthesised artworks. We then introduce a novel image representation that combines topological object structures and shape fitting. This abstract representation is used as a key component in our art generation framework. The type of art we produce are largely motivated by child drawing, cave paintings and these from artists such as Kandinsky, Mirö and late Matisse, all of whom organise simple elements such as shapes in a structured way to create life.

Part IV — Conclusions
Chapter 8 — Conclusions

In which we summarise the contributions that this thesis makes and offer a broad view on how algorithms proposed in this thesis justify our proposed contributions. Finally, we offer insights to where possible future works may lay and briefly discuss potential ways forward.

Chapter 2

Literature Review

This chapter reviews existing literature related to our research. The review is split into two main parts. The first part (Section 2.1) reviews the Computer Vision task of visual object recognition. Specifically, we first review the two independent yet related sub-problems of object instance recognition and object category recognition (or object classification). Special attention is paid to recent developments that act across depiction styles, i.e., techniques that are not limited to work on solely photographs. In the second part (Section 2.2), we review the current state of Non-Photorealistic Rendering (NPR) research. Particularly, we will highlight the recent trend towards producing abstract artworks and how high-level image processing techniques have been used to facilitate the production process.

2.1 Computer Vision: Visual Object Recognition

The earliest attempts to visually recognise objects can be traced back to the early 1960's. A famous example is the "Blocks world" [123] work of Roberts, where strong simplifications on both the object and scene were introduced. Figure 2-1 provides an example of such simple experiment setting. Over the last 50 years or so, the literature has grown from recognition of simple 3D objects to those depicted in natural image environments; and from finding specific instances of objects to category level recognition.

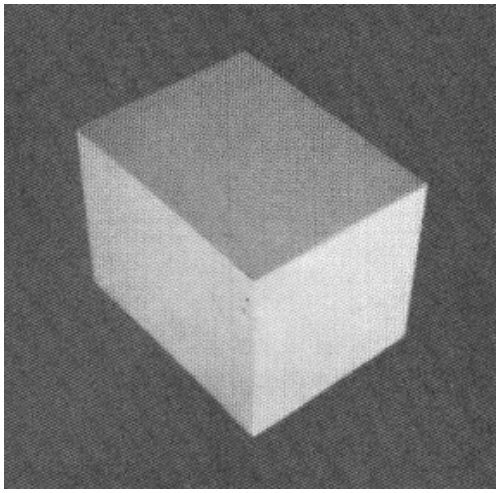


Figure 2-1: An example scene used by Roberts (1965) [123].

The traditional goal of visual object recognition is to find object instances or categories within a collection of images. This is a non-trivial problem, due to variances in object pose, lighting conditions, partial occlusion, to name a few. Consequently, the literature has progressed from being able to successfully address one of these problems to a combination.

Rather simple objects on an uniform background were studied by Roberts [123] in the 1960's. In the 1970's, research was mainly concentrated on range data, where 3D information can be directly used. The field then moved on to address object recognition on 2D natural images in the 1980's. However, work from this period could only recognise single object instances from a limited number of viewpoints. Category level recognition was also initiated during this period with the recognition of simple object categories such as digits and faces, again, with rather constrained environments. From the early 1990's, research has moved on to tackle recognition on a much wider range of objects and in much more complicated environments. However, to-date, work on visual object recognition has been mainly concentrated on photographs of objects and with some rare exceptions, typical algorithms tend not to generalise well to work on non-photorealistic depictions of objects, such as painting and drawings.

Despite the vast variety of object recognition systems proposed over the last 50 years, there is a strong commonality that is shared by all. A recognition technique normally relies on finding an invariant description of objects. It is eventually the robustness of such descriptions that determine that overall performance of the

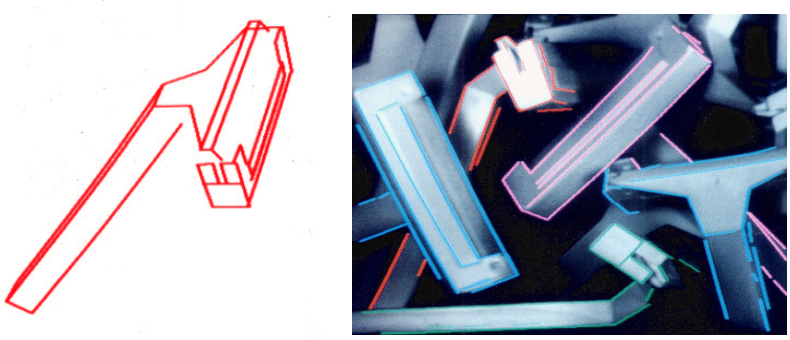


Figure 2-2: Left: 3D model of the probe object; right: successful matches superimposed upon the original image.

system. For example, if the description is invariant to illumination change, so will be the recognition system.

In the rest of this section, we will first offer a brief review of some representative approaches in object instance recognition in Section 2.1.1. Object category recognition is then reviewed in the following section (Section 2.1.2). Towards the end of this section, in Section 2.1.3, we will address recent development in the field of visual object recognition that acts across depiction styles. It is the latter body of work that this thesis contributes.

2.1.1 Object Instance Recognition

In this section, we will review the field of object instance recognition, i.e., finding specific instances of objects within images. Although not directly related to the core topic of this thesis, i.e., object classification, many traditional techniques developed to detect object instances are either directly used in object classification systems or have influenced their development.

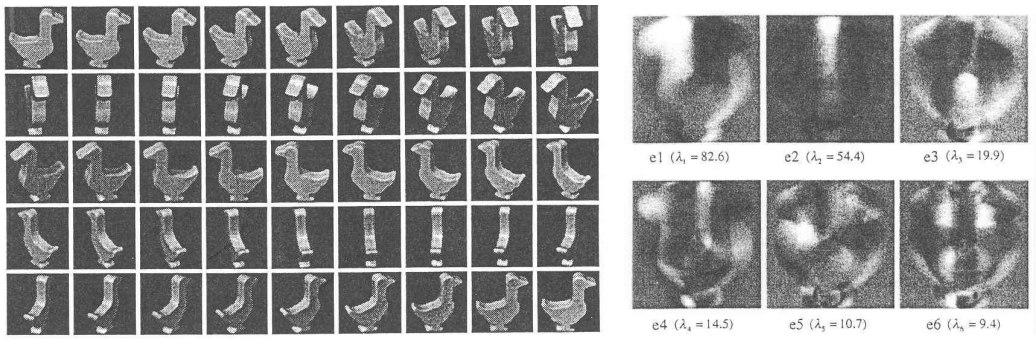
Geometry-based Methods

As previously mentioned, almost all visual object recognition techniques focus on finding a suitable object representation, which can then be matched. Early object instance recognition started by representing objects in terms of their geometric invariants. This was largely motivated by the availability of 3D object models. The most representative techniques that utilise geometric invariant are that of

Lowe [88] and Ullman and Huttenlocher [66]. Figure 2-2 illustrates an example of Lowe's [88] system. They both find instances of 3D objects by aligning 3D models with line segments found within 2D images; and because of the availability of full 3D information, such techniques could deal with significant change of object pose and partial occlusion. The problem of finding correspondences between the model and the target was however deemed as a particularly difficult one. On resolving this correspondence problem, Lowe [88] used the idea of perceptual grouping to form groups of line segments. The idea is that rather than searching for correspondences of individual line segments, which is computationally demanding; we can form groups of such and match on the basis on finding set correspondences, so that the correspondence search can be constrained. The idea of perceptual grouping also greatly influenced our first image description (Chapter 3); a detailed review of which can be found in Section 3.1.

Geometric hashing [78, 180, 126, 127] also received significant interest as an object recognition technique. These approaches maps geometric invariants of object models to an existing indexing table, which is pre-computed on a set of training images. Upon recognition, geometric invariant are first extracted from the probe images, which are hashed into the existing indexing structure; recognition is then achieved on a nearest-neighbour basis. There are several main advantages associated with the set of techniques related to geometric hashing: (i) rather than using image appearances as features, these techniques use geometric features as basis for recognition (ii) they are often invariant to a class of transformations, such as affine [78] and projective [126, 127] (iii) benefiting from the indexing structure, these techniques are often computationally efficient and scales sublinearly with the number of objects in the database, which is a crucial requirement for a practical object recognition system.

However, techniques that are based on geometric invariants face a few major drawbacks. The most significant of all would be that they all assume object contours can be readily extracted from images. However, in practice, extracting edges from natural images is often hard, due to different illumination, background and partial occlusion. Consequently, techniques that utilise geometric invariants of objects often assume objects pictured from uniform backgrounds, and are thus difficult to apply to images from more natural settings.



(a) (b)

Figure 2-3: Left: a subset of images captured to represent the object; right: the six largest eigen vectors that model the object.

Global Appearance Methods

Other than recognising objects based on their geometric invariants, researchers have also attempted storing "all" possible appearances of an object. In this fashion, recognition is reduced to asking the simple question of "Is the current probe object seen before?" This class of techniques towards object recognition are often referred as global appearance models. As the name suggests, this category of techniques model objects in terms of their global appearances; so that each object is represented as a large database of images captured under different conditions (varying viewpoints, different illuminations and so on). The work of Murase and Nayar [104] is a typical example. They densely sampled the viewing sphere of the object and stored images in terms of eigen vectors. Figure 2-3(a) demonstrates the type of images used in their experiment and Figure 2-3(b) shows the six largest dimensions in the corresponding eigenspaces. Another example is that of Schiele and Crowley [134]. Instead of using eigenspaces to represent images, the authors used histograms of local texture descriptors. Supervised learning techniques, such as Support Vector Machines (SVM), were also used to classify pairs of objects by Pontil and Verri [116]. In their paper, the authors mapped object images into a high-dimension feature space, prior to applying SVM.

Global appearance methods are often simple in nature. However, it is often difficult to sample an object under "all" varying conditions and even that is feasible, efficient storage and indexing of all possible instances of the object is inefficient. In addition, techniques based on global appearances of object also suffer from background clutter and occlusion, largely because of the global representations

used. Techniques that model objects in terms of their local features have been proven to deal better with such problems, as we will see in the following section.

Local Appearance Methods

In the previous sections, we briefly reviewed how objects can be represented in terms of their geometric invariants and how recognition can be achieved by representing images globally. In this section, we review another category of visual object recognition techniques that uses a collection of local object patches as basis for recognition.

This category of technique has proven to be successful in the last decade or so, largely due to the following advantages over other methods:

- Partial occlusions are naturally handled, resulting from the sparse nature of the object model, i.e., a collection of local object patches rather than the whole object.
- Background clutter can be better discriminated from the foreground object, when careful decisions are made on which local patches are selected for use in models.
- Invariance towards illumination and viewpoint changes can also be achieved using specific ways to describe the local patches.

In general, a local appearance model is built in the following two stages:

1. Feature detection: where local patches are identified on an input image. This stage is crucial for achieving robustness towards partial occlusion and background clutter. In order to obtain a robust object mode, we need many good patches on the foreground object as its representation.
2. Feature description: where the previously extracted local object patches are mapped into a feature space of some kind. This feature space often consists of dimensions corresponding to the photometric and geometric invariants found within the local patches. Collections of there descriptors together form the appearance model of objects.

(a) (b)

Figure 2-4: Example case of object recognition from the work of Lowe [89]. Left: three probe objects; right: probes detected in an image, where outlines of the object models are also shown.

Later on in this section, we will review feature detection and description techniques separately in more detail, as they are often used in object classification.

By representing objects in terms of their local features, the problem of visual object recognition is reduced to finding appropriate ways of extracting and describing local image regions. Upon recognition, a new object model consisting of many local object patches is extracted from an novel image, which is then matched into a database of existing models. However, in order to achieve robustness in recognition, several aspects of both the feature detection and description need to be carefully addressed. First, repeatability of the local features should be ensured across images of objects, pictured under many different imaging conditions. This is crucial to the overall recognition performance, as such feature patches eventually form the representations of objects. Second, region descriptors should be highly discriminative, i.e., patches that are visually different should appear further away in the feature space, whereas, similar patches should cluster tightly. Moreover, descriptors should be designed in a way that is invariant to changes in viewpoint, illuminations and so on. In practice, such feature spaces can often be designed for specific tasks. Lastly, there has to be sufficient number of feature patches on one object, so that not only the object can be more robustly modelled, partial occlusion can also be addressed.

To the best of our knownledge, the first to introduce the notion of local appearance models were Schmid and Mohr [135], who used which in the application of image retrieval. Their object models were also built by first detecting represen-

tative features on objects; which are then described in some invariant fashion. There is no novelty in their feature detection scheme; Harris interest point detector [60] were used to detect representative features on images. Circular patches around each Harris point are sampled, forming the basis for appearance model. Invariances of the object model towards rotation and scale were achieved through careful design of the feature descriptor. Rotational invariance was obtained by describing feature patch in terms of a set of gray-scale differential invariants; for scale invariance, they stored several circular patches around each Harris point at different scales.

Another well-know example that uses local appearance models of objects is the work of Lowe [89]. In that paper, the author proposed an object recognition system that is able to tackle not only scale, illumination and viewpoint changes; significant occlusion and background clutter can also be rather robustly handled. Instead of using Harris corner detectors [60], Lowe used extremas of Difference of Gaussian (DoG) operators to detect feature points on an image. As in Schmid and Mohr [135], Lowe uses circular patches centred around each DoG feature as local object patches to create the appearance model. However, the most important contribution in Lowe's paper is the proposal of "SIFT" descriptors, computable from these patches. For a local patch, "SIFT" descriptors samples image gradients within on a coarse spatial grid. Rotational invariance is made possible through computing descriptors relative to a dominant gradient orientation. A detailed review of the SIFT descriptors can be found later in this section, where it is reviewed along with other patch-based descriptors in the literature. Once the object model is built, recognition is performed in a nearest neighbour fashion, where the newly extracted model is matched into a database of already observed models. Lowe successfully evaluated his system on a decent sized database of objects of different poses, scales from rather cluttered backgrounds. Figure 2-4 offers an object recognition example of Lowe's work.

Both Schmid and Mohr [135] and Lowe [89] have successfully applied local appearance models of objects to the problem of image retrieval and object recognition, respectively. Their works have been proven invariant towards scale, rotation and changes in illumination. Despite their successes, both of their works can only handle changes in viewpoint due to similarity transformations, i.e., translation rotation and scaling. This is largely because of the circular local regions used to describe detected features. Figure 2-5 demonstrates the limitations of circular

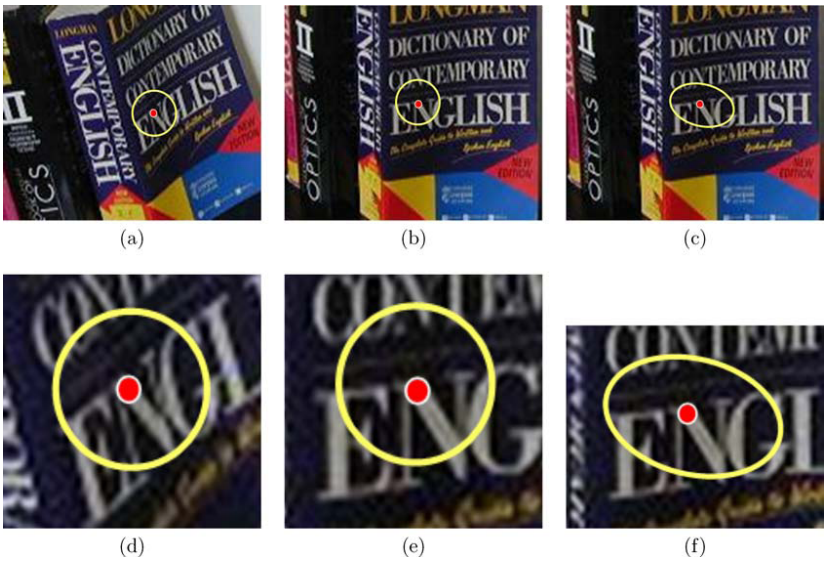


Figure 2-5: Figure demonstrating the limitation of circular support regions. (a)-(b): circular regions around the same feature point arose two different viewpoints. (c) ellipsal support region on the second viewpoint (b), that covers about the same area on the book as the circular region on (a). (d)-(e) closeups of (a)-(c).

support regions towards object viewpoint changes. Affine covariant regions were later introduced to solve this problem, which will be reviewed in detail in the following section.

Affine Covariant Regions

Affine covariant regions were introduced to tackle the limitations in early methods, which can only deal with viewpoint changes due to similarity transformations. They are called "affine covariant" because both their size and shape transforms in a covariant fashion with respect to 2D affine transformations. Although they are limited compared to perspective transformations, affine transformations do offer good local approximations to viewpoint changes, hence can be used as a good basis for local descriptor design. The consequence of affine covariant regions is that previous used circular regions now becomes elliptical; and corresponding elliptical regions across two viewpoints should cover the same area on the 3D object surface. Figure 2-5(d)-(e) also demonstrates the differences between circular region support and ellipsal region support.

An excellent review of affine covariant region detectors was conducted by Mikolajzcyk et al. [101]. In that paper, the authors conducted experiments using various

affine covariant region detectors to compare their performances against changes in viewpoint, scale, illumination, defocus and image compression. Overall, the range of affine covariant regions detectors can be categorised as follows:

1. Techniques that work iteratively to adapt to the actual shape of regions around feature points such as Harris corners, examples of such works include [4, 98, 133].
2. Techniques that find stable regions from threshold image intensities [93].
3. Techniques that fit shapes such as parallelograms and ellipses from interest points [166, 69].

There is no need for us to repeat the comparisons here; however, the work of Maximally Stable Extremal Regions (MSER) by Matas et al. [93] offers great influence to the first image description proposed in this thesis, hence is reviewed in more detail below.

MSER was proposed by Matas et al. [93] as a novel tool to tackle the wide baseline stereo correspondence problem. MSER works through thresholding the intensity image with all possible thresholds, $t \in S$, where $S = \{0...255\}$; regions that are stable across a sequence of thresholds are called stable and used as salient feature on an image. MSER produces a hierarchy of nested contiguous regions $Q_1, ..., Q_i$, where $Q_i \subset Q_{i+1}$. A region Q_i is said to be extremal if its average intensity is sufficiently different from that of its boundary pixels; and Q_i is called maximally stable if the following condition is met:

$$q_i = \frac{|Q_{i+\Delta} \setminus Q_{i-\Delta}|}{|Q_i|}$$

has a local minimum at i^*, where Δ is a tunable parameter, $\setminus$ stands for set difference, and $|.|$ denotes size of the region, i.e., the number of pixels within. It is worth to mention that MSER produces regions of arbitrary shapes, i.e., regions that correspond to image patches that are of extremal intensity. Figure 2-6 demonstrates successful MSER correspondences across a stereo image pair, where large viewpoint and scale changes are present. To compare it with other affine covariant region detectors, Mikolajzcyk et al. [101] used ellipses with the same second order moments to represent MSER regions. Their idea of stability is used

Figure 2-6: Top row: a stereo image pair with detected MSER regions and estimated epipolar lines overlaid; bottom row: closeups of MSER regions; the affine covarianceness is noticed by observing that corresponding regions cover the same area on the actual building.

in our first hierarchical image description explained in Chapter 3, where stable groupings are seeked in a perceptual grouping process.

Local Region Descriptors

As previously mentioned, there are two major components to a local appearance mode: feature region detection and description. Affine covariant regions offer reliable feature patches; the next question is how to appropriately describe such regions. Mikolajzcyk and Schmid [99] offered a thorough review of local region descriptors and conducted experiments to evaluate their performances. The design of region descriptions are essential to the performance of object recognition and classification, because they together form the abstract representation of the underlying object. A descriptor too exact would limit the degree of invariances towards illumination variations, scale changes and so on; on the other hand, a coarser descriptor would introduce many false positives in an object recognition system, for example. We will only summarise the set of commonly used descriptors below.

Probably the simplest way to represent an image region is to store its raw intensity

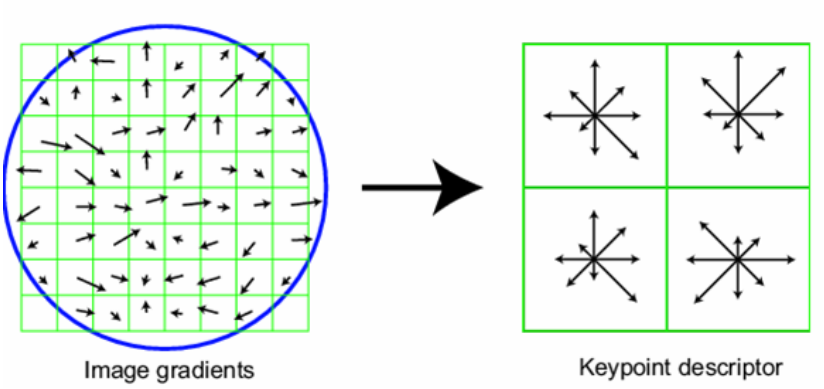


Figure 2-7: An illumination of Lowe's SIFT descriptor [46]. Image gradients are locally sampled and organised into a 4x4 grid.

values. The biggest problem with such descriptor lies with the fact that it does not offer any invariance to the description, hence strongly limits overall performance. Storage of such descriptors is also inefficient. Another way to describe regions is through the use of filter banks, such as steerable filters [46]. By doing this, instead of storing raw pixel values, only a set of filter responses is necessary. Rotation invariance can also be achieved by steering the filters relative to the dominant gradient direction within the patch [97]. Apart from using filter responses to describe regions, gray or colour histograms were also used [153].

Nonetheless, the most popular and commonly used region descriptor to-date is probably that of SIFT (Scale Invariant Feature Transform), proposed by Lowe [89]. SIFT is essentially a image gradient orientation histogram. Similar to previous methods [97], rotation invariance is made possible by computing gradient direction relative to a dominant region direction. Each descriptor typically uses a total of 16 histograms, which is aligned in a 4x4 grid. Gradients within each such histogram is categorised into 8 orientation bins. Overall, each descriptors is mapped into a 128-dimensional ($128 = 4 \times 4 \times 8$) feature vector. Figure 2-7 illustrates the image gradient histogram proposed by Lowe [89].

2.1.2 Category Level Object Recognition

In the previous section, we reviewed the traditional problem of object instance recognition. We have observed how researchers tackled problems brought by variances in viewpoints, illuminations, partial occlusion and so on, that often

appear on different instances of objects. Almost without exception, all the object recognition techniques reviewed so far work by first building an object model, which is then treated as the basis for the actual recognition stage. The one major difference among those techniques lies with the design of object models, e.g. whether it is based on geometric invariants of object or uses local object patches in a collective fashion. It is through the careful design of object models that we enable visual object recognition systems to perform in a invariant fashion towards change in viewpoint, illumination and so on.

Object instances recognition has come a long way, from working with objects depicted from rather plain background to achieving invariance towards changes in viewpoints, illumination, partial occlusion and background clutter. Despite such successes, the problem of object instances recognition is itself quite limited, in that the term "object" is often used in a semantic fashion. For example, in its most general definition, "chairs" means something that we can sit on. However, problem arises because there are thousands of chairs of rather different appearances, resulting from different styles, colours, materials and so on. The question then comes down to "how do we recognise a category/class of object called chairs?" Object category recognition is deemed as a harder problem than that of object instance recognition, exactly because of such intra-class appearance variations.

We as humans can easily recognise approximately 10,000 categories of objects using more or less equal efforts [6]. The early category recognition systems, however, started with recognising rather limited object classes, which are often relatively simple, i.e., relatively low intra-class variations. Examples of the early researched object classes include digits, faces, humans and cars, which we will review next. Afterwards, more recent techniques that tackle more general object classes are reviewed.

Categorising Simple Object Classes: Digits and Faces

Early research on recognising object classes started from the work on handwritten digit recognition. These techniques were motivated by industrial needs, such as automatic cheque processing and automatic US ZIP code reading. Handwritten digit recognition is regarded as an object category recognition problem rather than that of object instance recognition, because of the intra-class vari-

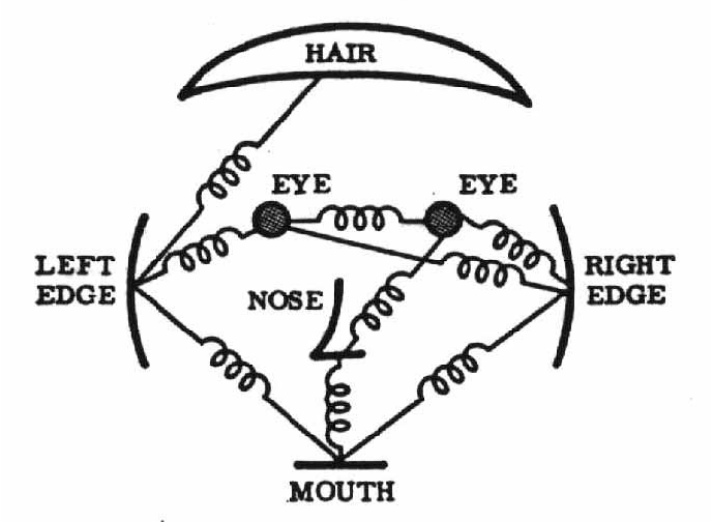


Figure 2-8: The parts and structure face model of Fischler and Elschlager [44].

ations in the appearances of each digit; as a result, each digit is treated as a visual class on its own. LeCun et al. [79] successfully proposed a supervised learning approach towards digits recognition. Their system is based on a trained convolutional neural network, which takes the segmented digit pixels as input and outputs a class label. Recognition is achieved in real time; however, like all supervised trained classifiers, it suffers from the significant training time required. Later, Weber et al. [173] successfully extended digit classification to that of Optical Character Recognition (OCR) using the concept of a constellation model, which we review separately. They did so without having to explicitly segment the characters from the background, an assumption previous methods often make [79].

Again, largely because of its practical value, face detection has been an active research field. It is important to differentiate face recognition and face detection; the former answers the question "whose face is this?", while the later says "where is the face". The famous "parts and structure model" was first introduced to detect faces, by Fischler and Elschlager [44]. As the name suggests, in this case, objects are represented as a collection of its parts and the geometric relationships in-between. Figure 2-8 shows the face model proposed by Fischler and Elschlager [44]. Although the overall performance of their system was quite limited, it is the concept of the parts and structure model that is important. We will see later in this section how this key idea has been used by other researchers.

One of the first successful attempts at practical-level face detection were due to Kirby and Sirovich [72], where the authors used Principal Component Analysis (PCA) to transform information extracted from face images to a low-dimensional space. Faces represented in this fashion are widely referred to as "eigenfaces". Other researchers also used the idea of dimension reduction to detect faces, either using PCA [165, 104] or techniques such as Linear Discriminant Analysis (LDA) [5]. However, one major downside of all above PCA-based algorithms is that they all assume the availability of face images pre-segmented from images, which is a difficult problem on its own. Moreover, because of their global nature, i.e., image-wide, they suffer all three major problems that the previously reviewed global object appearance models suffer, viz, sensitivity to illumination changes, background clutter and partial occlusion. Other machine learning techniques were also employed by researchers to tackle of problem of face detection, examples of which include the work of Rowley and Kanade using neural networks [128] and that of Viola and Jones using Adaboost [168].

Other than digit and face detection, researchers have also tackled more difficult object categories such as humans and cars, for example. Schneiderman and Kanade [136] perform wavelet transforms and use the resulting wavelet coefficients as models to be categorised. They demostated their system on both faces and side-views of cars. Human body detection was also tackled by researchers such as Papageorgiou et al. [112] and more recently, Dahl and Triggs [28] and Mikolajczyk et al. [100].

Simultaneous Recognition of More General Object Categories

Despite the successes in OCR, face detection and etc, with rare exceptions [136], all the above reviewed algorithms adhere to recognising specific objects of single categories. As a result, research on object categorisation have naturally shifted to designing system that are able to tackle many general objects at the same time. A category of techniques built on the concept of the "parts and structure" concept [44] prevailed here. Such recognition techniques include those based on "bag of words" model of objects and ones that use the so-called object constellation model. We will now review some representative papers in both algorithm categories.

The "bag of words" model were first used to represent objects in an object recog-

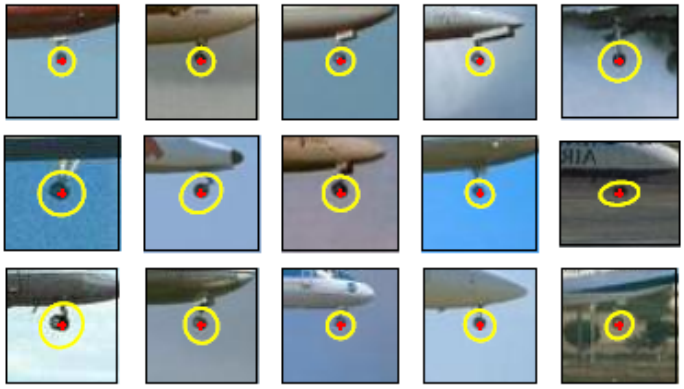


Figure 2-9: An example set of visual words — A wheel of an airplane. Figure reproduced from [150].

nition system by Csurka et al. [26]. In its most general definition, a "bag of words" model means a collection of "visual words", which collectively describe object categories. The term "visual word" stands for quantised local image patches around some feature points detected over a set of images. Figure 2-9 illustrates an example set of visual words. This is rather like the local object patches used in local appearance models for object instance recognition, reviewed in Section 2.1.1; the major differences lays with that "bag of words" models categories of objects, rather than their specific instances. The same set of techniques, such as Harris [60], DoG [89], etc, that were used to detect salient object features are also used here to detect features.

In the paper from Csurka et al. [26], the authors first extracted a large number of local image patches from a training set of images. This set of image patches then becomes the superset of "visual words", which are consequently clustered into a "visual vocabulary". This "visual vocabulary" is then used as the basis for image descriptions. Specifically, each image is described as a feature vector, containing occurrence counts in each cluster within the "visual vocabulary". A trained SVM from labelled feature vectors then becomes the classifier. Other authors have also used the "bag of words" model to enable object categorisation [108, 26, 150, 151, 149]; the major differences of these techniques lays with how classifiers were built. For examples, Opelt et al. [108] trained a set of weak classifiers and used AdaBoost to obtain robust ones, one for every object category. Having observed the need for heavy supervised training in many of the above techniques, Sivic et al. [150, 149] tackled object classification in a un-supervised fashion. Recently, Zhang et al. [185] published a comprehensive review on the subject of using local

features to classify objects, which includes an in-depth study on the "bag of words" model.

One major downside of the "bag of words" model is that it is unable to locate objects within images. This is largely due to the fact that the spatial locations of object features and the relationships among such are not modeled, which makes "bag of words" an "appearance only" model. Nonetheless, it is the lack of spatial information that made the object classification systems flexible in adding in more features, robust towards occlusion and etc. The key problem in introducing spatial relations to object models is how to incorporate such information without degrading system performances, while keeping the computational complexity low. Such spatial relationships that act to link up object parts were also an important aspect of the original "parts and structure" model proposed by Fischler and Elschlager [44].

Burl et al. [12, 13] and Leung et al. [82, 83] jointly introduced the notion of a constellation model of objects to detect faces. Different from that of the "bag of words" model, relative locations of objects parts are also modelled. This addition makes it a specific instance of the "parts and structure" model [44]. In their works, spatial arrangements of identified object parts are modeled in a probabilistic fashion; and face models were manually trained by clicking on pre-defined facial features across a training set of images. Because of the encapsulated spatial information, constellation models are able to infer occluded parts and better discriminate background clutter with the foreground features. Weber et al. [173, 171, 172] extended the work of Burl et al. and Leung et al., by making the previously manual training process automatic and making the model applicable to more general object classes such as cars. Fergus et al. [40] further extended the work of Weber et al., by offering a more rigorous approach that is entirely probabilistic. They modelled both the appearances of parts and their relationships as Gaussian distributions. As a result, their system is able to yield a probability for each match. In order to demonstrate the superiority of their technique, as oppose to that of Weber et al., they evaluated their object detection system on a much wider range of object categories, ranging from faces to motor bikes. Figure 2-10 illustrates a constellation model of a motor bike used in their experiment, together with its spatial shape model and quantised local features.

Others have since extended the use of constellation models to specific problems

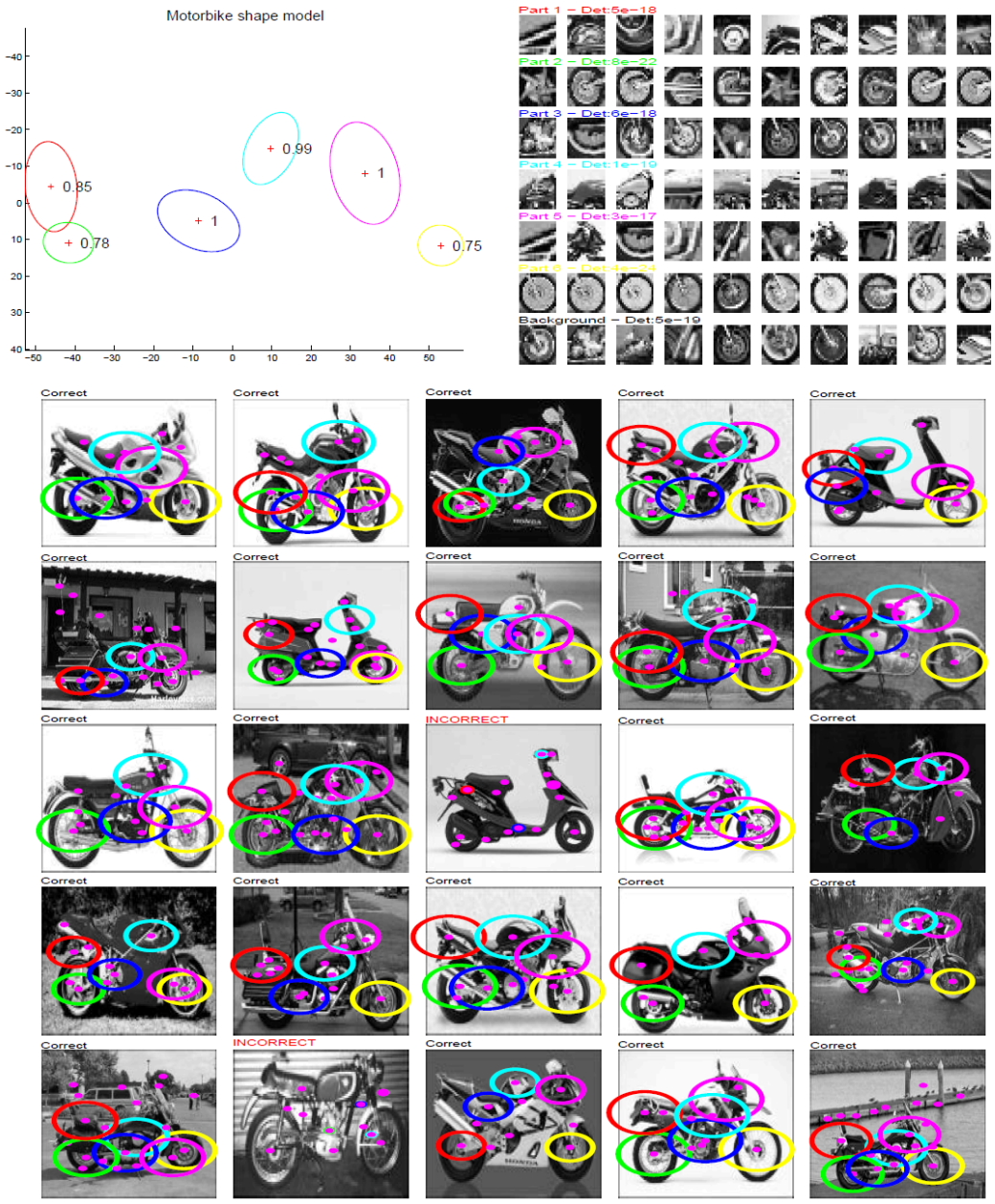


Figure 2-10: Top left: the learnt constellation model of a motor bike; top right: some example visual words; bottom rows: fitting results of the constellation model to a few testing images. Figure reproduced from [40].

such as human detection and pose estimation, independently by Felzenszwalb and Huttenlocher [39] and Ramanan et al. [121]. An important contribution of Felzenszwalb and Huttenlocher [39] lays with its low computational complexity, achieved using an efficient tree structure to model the spatial relationships among parts. Crandall et al. [25] explicitly studied the relationship between complexity of the parts model with the computational cost of the recognition system

and published results in favour of rather simple parts models. Fei-Fei et al. [37] successfully shown through the use of constellation models, that it is possible to train object categories using very few sample images, typically less then five. They obtained such results by introducing a novel hierarchical Bayesian version of the constellation model. Leibe and Scheile [81] successfully proposed an object categorisation system that is capable of recognising a large set of objects; Torralba et al. [162] then further extended the range of recognisable object categories by demonstrating decent performance on 21 object categories, however, they argue that the proposed "joint boosting" technique is able to tackle hundreds of categories. Researchers have since proposed many object categorisation systems [51, 105] based on the idea of using "parts and structure" models to describe objects; the essence of these techniques are quite similar, it is often the details on model building and classifier training that are different.

All the above methods use local image patches as basis for object parts, researchers have also used contour fragments to model objects, examples of which include [76, 145, 109, 41, 146]. Similar to object instance recognition techniques that rely on geometric invariants, such methods often share a common downside that extraction of true edgels are often hard in natural settings.

In summary, the problem of learning objects classes covers a considerable body of literature, which the above review only partially covered. We have observed that much of the successful work is based on constellations of local features. These features often come in the form of image patches, but lines and curves are used as features too. There is a common idea underlying all of these approaches. It is to use features construct a vocabulary of visual words, and then combine these words to make objects. To construct a vocabulary putative words are isolated using a feature detector which is robust to translation, scale and affine variation. The centres of these feature patches can be identified using interest points [60] or as extrema in a difference of Gaussian filter [89, 99]. Patches can be described in several ways, with SIFT [89] being the most common. The end product is the same in any case, which is a set of patches over an image, each patch being described by fixed-length feature vector. Patches like this allow image pairs to be matched via a similarity function of some kind. Given such a similarity measure, classifiers can be constructed using some standard approach, support vector machines and N-nearest neighbour classifiers are in common use. Given a set of images known to belong to one class, the features shared by class

elements can be identified, and these become the words upon which a vocabulary is based. Semi-supervised training, in which images are somehow labelled with class identity is often used.

2.1.3 Cross Depictive Styles Object Recognition

In the previous sections, we have reviewed the traditional problems of object instance recognition and object category recognition. We have observed that both fields are rather interleaving than independent of each other. Many techniques developed for object instance recognition have been successfully applied in object classification and vice verse; techniques aimed at solving the problem of classification can also be used in the former case of object instance recognition. Particularly, we have seen how the notion of the "parts and structure" models of object have been successfully applied in object classification — notably in constellation models.

However, in this thesis, we are interested in classifying objects no matter how they are depicted; so we want a classifier that works equally well on not only photographs of objects, but also paintings and drawing of such. The previously reviewed methods based on "bag of words" models that have found lots of successes in categorising hundreds of objects would fail in this case. We argue that the use of literal words is both the strength and the weakness of these approaches: a strength in that it allows a great many problems to be solved and so supports a great many applications; a weakness in that it restricts words to encompass just one depictive style. This is because the words in such a vocabulary are literal in the sense that they represent the appearance of some part of an object. This is true no matter whether patches or contours are used. Patches capture local colour distribution whereas contour fragments capture local shape. The limiting assumption is that the visual words exhibit low variation, and since the words are of a literal nature this places an in-principle restriction upon the gamut of images any such vocabulary can describe. Since photographs are almost invariably used, this gamut tend to be restricted to photorealistic images. Indeed, some deliberately filter non-photographs from their database used to build models of appearance [137]. Thus objects depicted in a style other than photographic cannot be recognised.

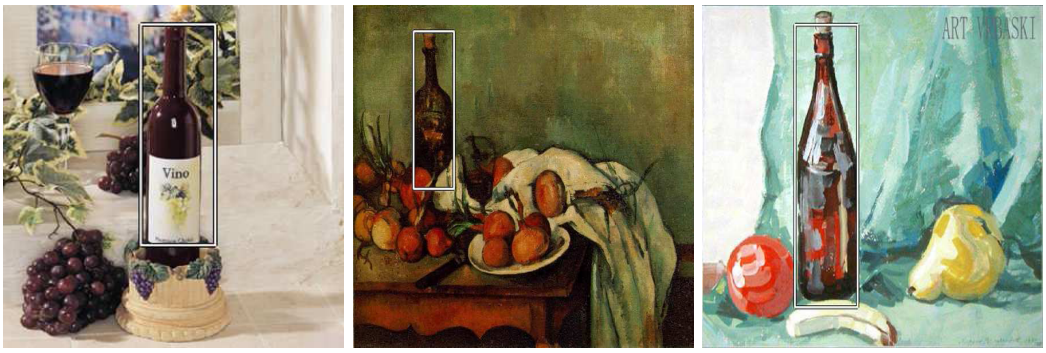


Figure 2-11: A wine bottle detection result of Ferrari et al. [41]. Note the successful detections on both photographs and paintings.

Relatively little research energy has been expended on classifying objects independently of depiction type. Classifiers using geometric invariants [88, 66] and curves [76, 41, 146] as the basis of class identification are typically motivated for robustness to variations in lighting, colour texture, and clutter. It is true that such systems may admit a wider class of depictive styles than patch based systems. Indeed, Ferrari et al. [41] use hand made line drawings to detect objects in the presence of clutter. Despite being able to detect regardless of depiction, their system was evaluated on a rather limited database of five objects over 255 images. Figure 2-11 demonstrates successful detection of wine bottles within both photographs and paintings, using the work of Ferrari et al. [41].

Other have also addressed the problem of matching line drawings or colour sketches to photographs, especially in the field of Content-Based Image Retrieval (CBIR) [29]. Jacobs [68] employed wavelet transforms to map images into their signatures, which are essentially vectors of the few largest wavelet coefficients. Given a colour sketch, a new signature was calculated in the same way and matched into the existing database of image signatures. A recent innovation along the lines of querying using sketch-based examples is that of Chalechale et al. [15]. The authors used binary line drawings to query into a database of colour images. Upon retrieval, a similarity score was computed between a full colour image and a simple black and white sketched query; a measure that is based on strong edges extracted from the colour image and the morphologically thinned outline of the query image. Overall, such approaches often emphasis on first extracting a set of main characteristics buried within the line drawings or colour sketches, which are then treated as features that are searched for within images. This is rather different from a classifier which is based on some abstract class

Figure 2-12: A collection of rather different depictions of a heart, which can be successfully classified using Schectman and Irani [141]. Figure reproduced from Schectman and Irani [141].

properties and is operational across depictive styles, a literature gap we seek to fill.

Fidler and Leonardis [42] are also able to match across depictive styles. They use Gabor filter responses at different scales as words. These responses are combined in hierarchical fashion, into ever more specific objects until the highest layers are category-specific. The upper layers of the hierarchy require user supervision. The system is then able to detect across depictive styles because it is premised on simple filter responses rather than patches.

The problem of matching images in different depictive styles is directly addressed by Schectman and Irani [141]. They use the spatial relation between different patterns, which they call local self-similarities. Their insight is that matching a pair of objects requires only consistency between matched parts. So, for example, if in one image flowers represent eyes they can be matched to the photographic eyes in another picture, or the drawn eyes in yet another picture. Since this provides a similarity measure, it may be possible to use such a matcher as the basis for a classifier that crosses depictive boundaries. Such a classifier would operate by matching a given image to class exemplars, at least one would be required. Figure 2-12 shows successful matches of different depictions of a heart. Such an approach is adopted by Frome et al. [47], who use what they call focal images and a single deformation to specify a visual class.

The work in this thesis is closer in form to that of Ahuja and Todorovic [160] than any other mentioned so far. They match on the basis of an objects parts, taking in account geometric, photometric and topological properties. They match via tree isomorphisms. Graph matching has received a great deal of attention, and Ahuja and Todorovic [160] provide an excellent review we need not repeat here.

We too match in a graph theoretic fashion, as will become clear in Chapter 6 of this thesis, where we attempt the problem of classifying objects regardless of their depictive style.

In summary, standard approaches using patches are premised on visual words of low variance across images. This assumption is violated when the image set includes a variety of depictive styles. Many recent works advocate the use of contours, and these can detect given exemplars in images of different kinds. But this restricts one to contours, also general line drawings correspond only weakly to the derivatives of photographs [58]. In this thesis, we show that matching between images in different depictive styles is possible without using contours, but using objects' structural information.

2.2 Computer Graphics: The Trend Towards Abstraction

This section offers an up-to-date review of the field of Non-Photorealistic Rendering. In particular, we will highlight the trend towards producing increasingly more abstract art that has developed over the last 20 years or so. However, at first, we need to differentiate the field of Non-photorealistic rendering from photographs (NPRP) from non-photorealistic rendering from three dimensional models, which also received considerable attention over the past couple of years. Our contribution is to the former — so we cite none of the latter, but only to better focus the review and highlight the contributions of this thesis.

NPRP research was pioneered in the early 1990s with interests in brush modeling [115, 118] and media emulation, such as watercolour [152] and pencil [18]. Shortly afterwards, the desire to built fully automatic painterly rendering systems became an active research area. Such trend started with several semi-automated paint systems [55, 130] and ended up with works that are able to automatically render images into impressionist paintings [84] and oil paintings [61], for example. Researchers then realised the need to abstract, something art itself finds its essence in. Abstraction in NPRP was first introduced as ways to paint salient objects in more detail [19, 21]. The trend towards abstraction was then continued in many ways [3, 7, 111], with the needs of increasingly higher-level information

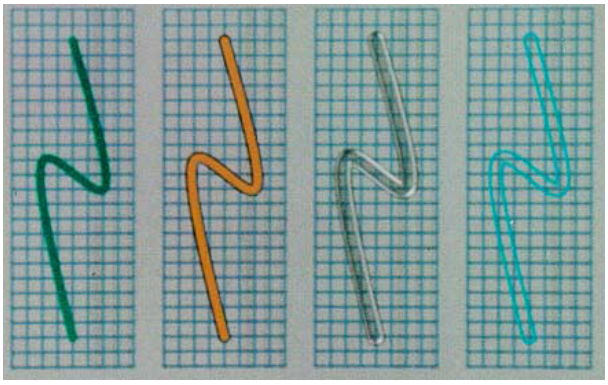


Figure 2-13: Brush strokes produced using Whitted's "stamping" technique [176]. Figure reproduced from [176].

from images, often extracted using techniques in the Computer Vision literature.

In accordance with trend in the chronology of NPRP, the review will be divided into three sections. In Section 2.2.1, we will review the early attempts in NPRP that builds brush models and simulate various types of artistic medias. The push towards fully automatic painterly rendering systems is then reviewed in Section 2.2.2. Finally, in Section 2.2.3, we will highlight the later trend towards abstraction from the field of NPRP, which this thesis too follows.

2.2.1 A History of Brush Modelling and Media Emulation

The early NPRP literature has been populated with attempts to model a brush. Whitted [176] first introduced the notion of a brush stroke. In his work, a brush was simply represented as a 2D pattern. In order to create a stroke, the brush is "stamped" across the image following a pre-defined path. Because of the nature of his approach, i.e., a repeation of "stamps", brush strokes often look rigid. For instance, strokes have constant width and identical appearance along the path. Figure 2-13 illustrates the kind of brush strokes Whitted was able to produce.

Strassmann [157] was among the first to model brushes in terms of their physical properties. Each brush is modelled as a 1D array of bristles, which is then swept along a 2D spline to create a stroke. The rigidness seen in Whitted's [176] strokes were tackled in two ways: (i) each bristle was assigned a value indicating how much paint is left, hence creating variances in the appearance of the stroke along the path; (ii) pressure could be incorporated to stroke paths, which enables

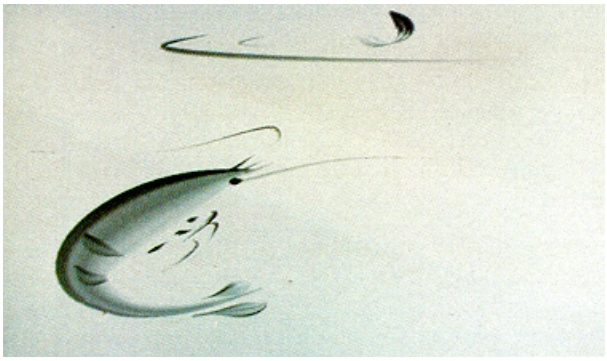


Figure 2-14: An example sumi-e rendering of [157].

strokes to have varying widths. Using his "hairy brushes", Strassmann was able to produce quite convincing sumi-e style renderings, Figure 2-14 offers an example of such. A later attempt on producing sumi-e rendering also belongs to that of Pham [115], whose approach is similar to that of Strassmann [157], but strokes are scan-converted in a single pass.

Pudet [118] studied how can brush strokes can be rendered in real-time, so to enable interactive painting. He made use of a cordless stylus device as a brush. Similar to Strassmann's [157] "hairy brushes", stroke appearances are dependent on their positions and pressures. In addition, he also used the angle of the stylus to change the width of strokes.

Lee [80] offered an in-depth study on modelling the physical brush, after observing the lack of rigorous research on physical brush modelling from previous attempts [157, 115]. As a result, he proposed a 3D brush model that incorporates both the friction and inertia of brush bristles. Xu et al. [184, 183] went even further in the direction of physical brush modelling, by proposing a volume based brush design. In addition, the interaction between brush bristles and the substrate was also fully simulated. The authors presented some impressive results producible using their system, especially on Chinese calligraphy; an example of which can be seen in Figure 2-15.

All the above reviewed methods were inspired by building a physical model of painting brushes. However, strokes produced this way are either un-realistic [157, 118], or demands lots of computing power [184, 183]. Instead of modelling physical brushes, there are other techniques that aim to mimic the appearances of artistic strokes, to the extend that they are realistic and aesthetically pleasing

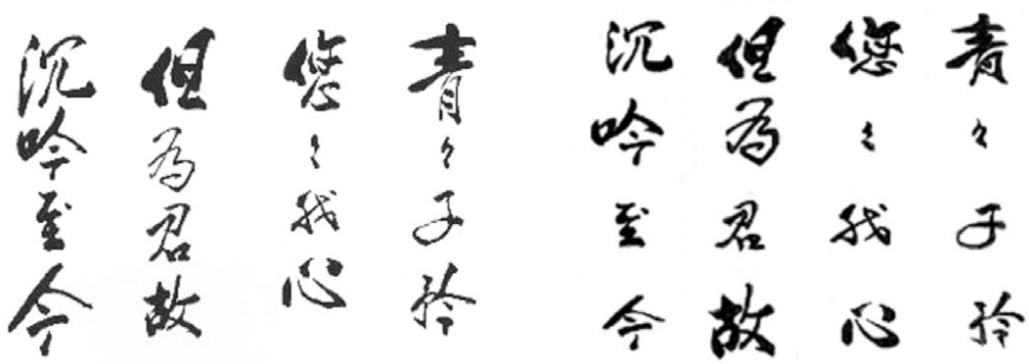


Figure 2-15: Chinese calligraphy produced by Xu et al. [184, 183]. Left: real artwork; right: imitation. Figure reproduced from [184].

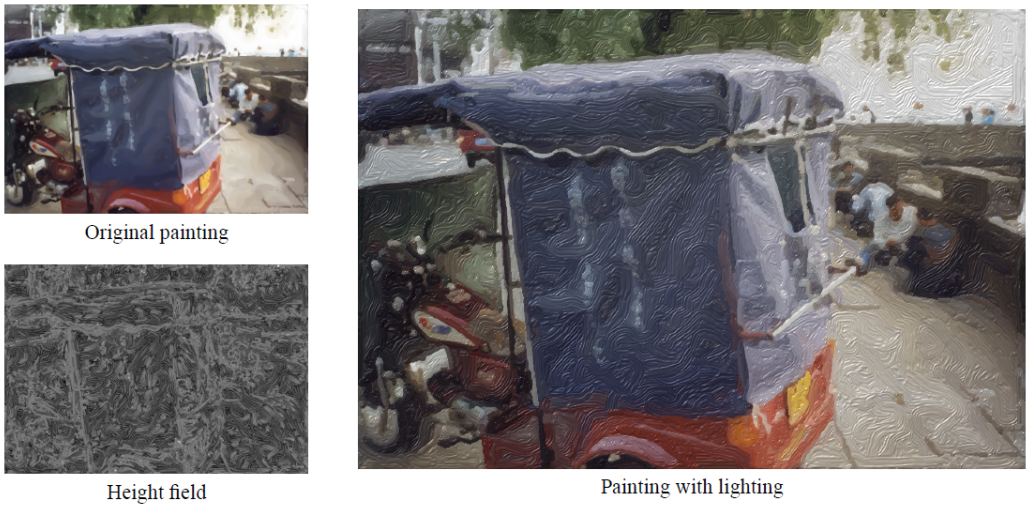


Figure 2-16: An oil painting rendering from the work of Hertzmann [63].

to look at. Such category of technique mainly rely on being able to texture map the appearance of an example brush, i.e., a texture template sampled from real strokes, onto a stroke trajectory. In this way, many types of artistic media, such as crayon and paint, can be emulated.

The first work that brought the concept of using texture mapping into making artistic stroke renderings was from Hsu et al. [65]. In their paper, the authors presented "skeletal strokes", which use arbitrary pictures as brushes, as oppose to "stamps" or bristles. Hertzmann [63] also proposed a simple yet powerful way to render strokes using texture mapping together with bump mapping. In addition to the normal stroke rendering stage, a bump map is also created by rendering brush strokes textured with the height maps, which is then used to

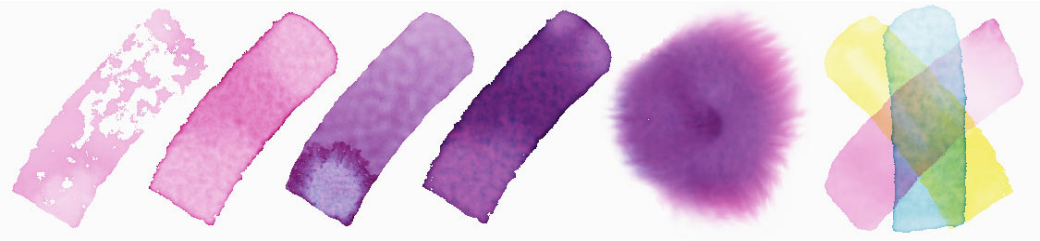


Figure 2-17: Watercolour effects produced using the work from Curtis et al. [27].

relight the original rendering. Figure 2-16 illustrates an oil painting rendering using Hertzmann's [63] method, together with the corresponding bump map. Other advantages of our measure over Hertzmann's work include its flexibility towards rendering styles and its computational efficiency.

Despite the successes in creating rather aesthetically pleasing brush strokes, all the previously reviewed methods share a common downside, that is brush strokes tend to be of single scale, which poses the problem of pixelisation once large-scale magnification is introduced. Perlin and Velho [114] proposed a multi-resolution painting system that stores strokes either as procedural textures or band-pass pyramids. Upon display, brush stroke textures are pre-computed, so that pixelisation effect is effectively avoided.

In order to create more visually pleasing rendering effects and extend the gamut of NPRP styles, authors have also studied the interactions between artistic media and substrate. An early example is from Small [152], where he modelled the effect of applying watercolour brushes on paper. The flow of watercolour on the substrate is modelled in terms of a 2D array of automatas, each of which corresponds to one pixel. Absorbency of paper was achieved by introducing a novel paper model, made up from intertwined paper fibres.

Curtis et al. [27] also tackled the problem of simulating watercolour effects. The authors proposed a complex multi-layered model of the substrate, using which they are able to better simulate the pigment-substrate diffusion and to produce wet-on-wet effects through a bi-directional transfer process. Figure 2-17 illustrates the kind of watercolour effects Curtis et al. [27] were able to produce.

Another well-known painterly rendering system that is able to emulate a wide range of artistic effects is due to Cockshott et al. [18]. Their system is able to model both wet and sticky media on canvas. The design of the system is rather

complex, however, its major contribution lays with the use of height field which is bump mapped to create a visual effect of build-up paint.

Other than creating painterly artistic effects, researchers have also worked on emulating other medias, such as the graphite pencil work by Sousa and Buchanan [154, 155, 156]. In their line of works, the authors introduced a physical model of a pencil based on its hardness and the shape of the top. Relatively realistic 2D line drawings were presented to demonstrate the quality of their pencil emulation. Takagi et al. [159] extended the look of pencils by incorporating colour information. Recently, Meraj et al. [95] proposed a simple system to create human-like pencil drawings. The authors proposed a novel pencil texture model based on co-occurrence matrice of intensities values from sampled pencil lines. Special attention was paid to model the line trajectory that conforms to human arm movements.

O'Donovan and Mould [106] recognised the lack of media simulation other than the traditional oil and watercolour. They presented an interesting system that is able to render images as if they were produced by a felting process. The felted style is achieved by computationally modelling each stage in an actual felting process. Recently, Brooks [9] studies how mixed-media artwork, which incorporates two or more traditional artistic media, can be rendered from images. The proposed system is based on the concept of seamless mixing of a set of traditional NPR filters. It works by first building an image hierarchy of region segments, each level of which had a specific NPR filter assigned. Finally, separate renderings of different levels are then merged by solving Poisson equations. Other authors have explored the various novel means to paint, such as the very recent "Real-time gradient-domain painting" work by McCann and Pollard [94]. In their work, a novel gradient-domain brush strokes was proposed and benefiting from a fast integrator and a simple multigrid algorithm implemented in GPU, rendering can be achieved in real-time. The system was demonstrated in two applications, image editing and synthesising art, as Figure 2-18 illustrates. Due to the nature of their gradient brushes, artworks produced are often of abstract style. It is however important to note that all artworks presented were created by artists, instead of automatically synthesised from photographs.

(a) (b)

Figure 2-18: Two novel applications using the gradient-domain brushes by McCann and Pollard [94]. Left: image editing, note the new window; right: an example painting created by an artist using gradient brushes in around 30 mins.

2.2.2 Towards Fully Automatic NPRP Systems

The search towards fully automatic NPRP systems started with several systems that required some degree of human interaction. Later, increasingly complicated Computer Vision algorithms were used to facilitate the automation process.

The landmark paper, "Paint by Numbers" [55] by Haeberli, acts as the first attempt in the search for automatic NPRP systems. In that paper, Haeberli proposed a semi-automatic painting system, using which users can create impressionist style renderings. The system works by a user repeatedly clicking on a reference colour image and each click draws a stroke on an empty canvas. Upon each click, the location, orientation and colour of the corresponding stroke is sampled from the reference image. The width and length of the strokes can be adjusted by the user to reflect the salience of parts on the reference image. Despite involving a certain amount of user interaction, the system is very easy to use and can often be used without prior training. Figure 2-19 shows some paintings produced using Haeberli's "Paint by Numbers" [55] system. Another important contribution from Haeberli [55] is the notion of treating a painterly rendering as an ordered list of strokes. This concept was used in many following papers that aim to produce stroke renderings [27, 63].

Instead of interactively making impressionist paintings, Salisbury et al. [130, 129,

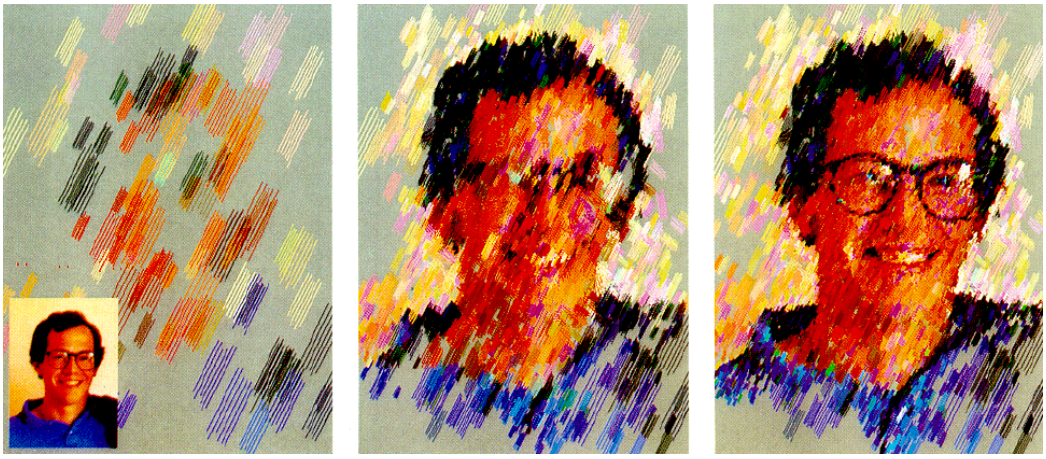


Figure 2-19: Some example renderings using Haeberli's "Paint by Number" [55] system. Figure reproduced from [55].

131] have studied the problem of creating digital pen-and-ink illustrations. Their system is backed by a bank of textures which can be selectively mapped to strokes. At the same time, they also used low-level image information such as edges to clip textures. Interactive stippling systems were also an interest of various authors [31, 11]. The idea of a stippling system lays with representing images in terms of stipples, which are rather like how strokes are used as rendering primitives.

All the artistic rendering techniques reviewed so far in this section requires some amount of human interaction. Although human input can lead to pleasant looking artworks, the process is often labour intensive and time consuming. The research in NPRP has naturally moved to concentrate on making full automatic systems that are able to automatically render images into different artistic styles. In particular, the concept of representing a rendering as an ordered list of strokes, first proposed by Haeberli [55], have been intensively used here. Importantly, strokes can have different forms to create renderings of various artistic styles. Based on such concept, when designing an automatic NPRP system, two main questions should be answered: where to put a stroke and what the stroke should look like. In some cases, the ordering of the strokes is of some importance as well.

An early example in such direction was that of Haggerty [56], who extended Haeberli's "Paint by Number" [55] system to be automatic. However, the rendering results using their automated system are often quite poor, due to the random processes introduced to set parameters in Haeberli's system.

The first automatic, image-dependent painting system was due to Litwinow-

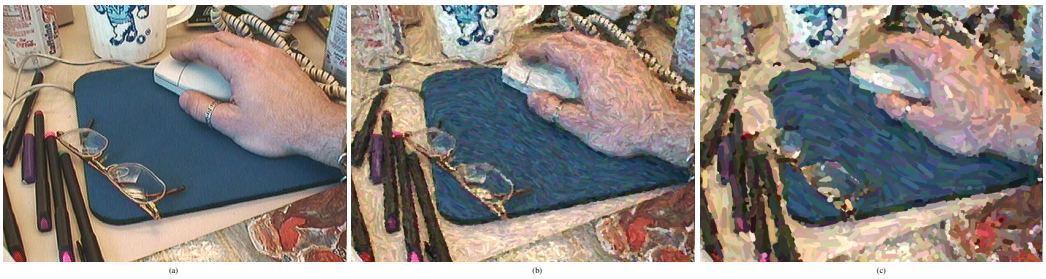


Figure 2-20: Figure demonstrating two rendering results using Litwinowicz's [84] system. Left: original colour image; middle-right: rendering results with two different settings for stroke widths.

icz [84]. The automation process is supported by an underlying image gradient field, extracted using Sobel filters. The placement of strokes is constrained by this gradient field; for example, strokes are orientated to the tangents of local gradient fields and any stroke that goes across a thresholded edge map is clipped. However, one major drawback of Litwinowicz's [84] is that strokes are rendered in a random order, which makes the rendering look rather noisy. Moreover, due to the fixed length strokes used, the background and foreground objects appear in the same level of details. Figure 2-20 demonstrated such problems in Litwinowicz's [84] renderings. Treavett and Chen [164] and Shiraishi and Yamaguchi [143] also tackled automatic paint systems that employed statistical measures to place strokes. For example, Treavett and Chen [164] aligned randomly placed strokes with their principal axes, obtained using eigen analysis on pixel intensities.

Hertzmann [61] importantly realised that artists do not paint in the same degree of scale, but use finer strokes to highlight salient entities in the scene and coarser ones to paint the background; a problem observable from Litwinowicz's [84] renderings. Hertzmann proposed a simple multi-scale and multi-layer rendering algorithm to tackle this problem. Each layer is painted in different degree of details, according to its corresponding scale. Specifically, coarser strokes were applied on layers of a large scale and rendered before finer ones found on lower scale layers. Moreover, finer strokes are only rendered if its appearance is sufficiently different than the coarser stroke underneath. Hertzmann also proposed ways to create long and curved strokes based on β-splines in the same work, which dramatically increased the aesthetic quality of the paintings produced. Figure 2-21 illustrates Hertzmann's [61] work in separate stages. Gooch et al. [50] also proposed a painting system that utilising curved strokes. Instead of using β-splines to model stroke trajectories [84], the authors fitted strokes to medial axis computed from

Figure 2-21: Separate stages of Hertzmann's [61] work. Top left: original colour image; top right and bottom left: two layers of paintings, from coarse to fine; bottom right: the final rendering. Figure reproduced from [61].

regions of homogeneous intensities.

Collomosse and Hall [19, 21, 22] continued the trend in producing paintings that reflect the salience of different entities in an image. Instead of using local and low-level image information such as edge maps [84, 61], they analysed images in a global fashion using higher-level computer vision techniques and machine learning algorithms. The use of global image information in NPRP systems acts as a major contribution from these authors. For example, in [22], the authors trained a classifier on a few training images, which was then used to classify pixels, hence produce a salience map. The whole rendering process is guided by a relaxation process based on a novel genetic algorithm. This relaxation process results in a fine to coarse separation between salient and non-salient regions. An example rendering is shown in Figure 2-22; it can be seen that salient image details are covered by finer strokes than the background, such as the head of the dragon. Apart from computing salience maps from images, DeCarlo and Santella [30] also cleverly used eye-tracking data to induce salience maps and create a manually driven process with a low cognitive load on the user.

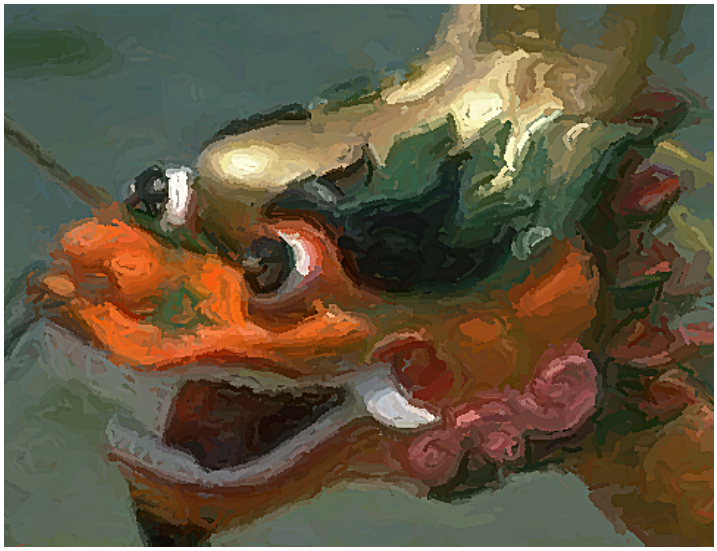


Figure 2-22: An examples rendering using the genetic painting system of Collomosse and Hall [22]. Figure reproduced from [22].

Other authors also used relaxation as means to produce computer-generated artworks, Hertzmann's [62] being the foremost runner. In his work, an objective function was used to seek a "best" painting, i.e., one that minimises the objective function. The system works in an iterative fashion. Upon each iteration, a score was computed from the current state of the painting using the objective function; and the iterative process terminates when a "good" score is found. The objective function is crucial to the quality of the final rendering and in the case of Hertzmann [62], it is designed to perserve high frequency information within an image. A similar objective function design concept was also taken by Sziranyi and Tath [158], where the authors also proposed a relaxation based NPRP system.

Instead of controlling the specifics of a painterly rendering system using a predefined set of parameters, Hertzmann's [64] presented an creative way of controlling the artistic styles of painterly rendering systems, called "Image Analogies". In his work, Hertzmann used real paintings to control the artistic styles of the renderings. More specifically, given an image A and a painting of it A', the aim is then to copy the style captured in the painting A' to create a painting B' of a new image B. Hertzmann proposed a rather simple yet effective way to do just that; for every point on B', its value is copied from A', where a best match exists between A and B.

Shugrina et al. [147] presented an automatic NPRP system that utilises human

emotion states to control the look of the final renderings. Their system is able to map facial expressions of human into a pre-defined set of rendering parameters, such as stroke colours, width, length, the jaggedness of its path and so on. This set of parameters are then fed into a rendering engine to produce the final output. For example, a painting would have brighter colours and smoother strokes when a happy face was presented; and vice verse when the system saw a sad face.

2.2.3 Moving Forward to Synthesising Artworks of Abstract Styles

In Section 2.2.1, we have reviewed the history of brush modelling and media emulation, when researchers seeked to answer the question of "what does a stroke look like?". The work is of great importance because stroke rendering is commonly used by NPRP systems, ranging from the automatic to interactive and all in-between. Later in Section 2.2.2, we have observed that while designing fully automatic NPRP systems, the question of "where to place strokes" was seen as an open question. It became clear that low-level image processes such as edge-maps [84] are not sufficient, because these act locally. Salience maps [21, 22] allowed the location of each stroke to be decided on image-wide (global) information.

However, it was not until only recently have the means to extend the gamut of NPRP to abstract styles been recognised. For example, Mould [102], and later Setlur et al. [139] synthesise stained glass renderings. Collomosse and Hall [20] cut out, re-arranged and distorted segmented image regions to create Cubist-like renderings from photographs. The same authors have recently shown how to introduce non-linear (artistic) perspective into photographs [57]. Others reflect the intuition that art making depends on inferring perceptual structure from images so as to facilitate their rendering [111, 77]. In the rest of this section, we will review this recent trend towards introducing abstraction into NPRP in more detail.

The first push towards synthesising abstract art was motivated by the use of image segments, often obtained using image segmentation techniques found in the Computer Vision literature. This trend seems natural as image regions are of much higher level primitives than strokes, hence act as a better basis for abstract

Figure 2-23: Two renderings of different artistic styles from Bangham et al. [3]. Left: a watercolour rendering; right: rendering in the style of cross-hatching. Figure reproduced from [3].

renderings. An early example of using image segments as rendering primitives was the work of Bangham et al. [3]. Using image sieves, for a given image, the authors were able to construct a hierarchy of image segments, where each level of the hierarchy corresponds to segments at a given scale. This hierarchy was then analysed to yield a covering set of salient regions of different scales. Finally, traditional NPRP filters were applied to produce artistic renderings, as Figure 2-23 shows.

Despite the use of image segments, renderings produced by Bangham et al. [3] can still look quite photographic, i.e., the abstractness in their renderings are quite limited. Probably the first work that really extended the gamut of NPRP styles in the direction towards abstraction, was from that of Collomosse and Hall [20]. These authors presented automatic means of rendering images into Cubist-like paintings. They used regions cutouts from images taken from different perspectives of the same scene; feature regions were then selected based on a novel salience measure. Finally, these salient regions were composited to make Cubist-style paintings. Figure 2-24 illustrates a couple of such paintings.

Mould [102], Setlur et al. [139] and Brooks [8] all studied the problem of generating stain-glass renderings from images. All these techniques rely on the ability

Figure 2-24: Two Cubist-style renderings produced using the system of Collomosse and Hall [20]. Figure reproduced from [20].

to segment images into large and coarse regions. For examples, Setlur et al. [139] replaced segmented regions with translucent texture patches. Manual interaction are often needed to correct segmentation errors and merge smaller regions into large ones. Once a desired segmentation is obtained, the content of each segment determines the appearance of glass shards by visually querying a texture database.

Wen et al. [174] also used means of image segmentation to produce visually pleasing colour sketches. In their work, an image was first segmented; and just like previous works in stain glasses [139, 8], the segmentation map was manually edited by putting down scribbles to obtain large regions that correspond to semantic entities in the scene. These regions were then shrunk via morphological erosion and coloured to produce colour sketches. Figure 2-25 illustrates some example colour sketches produced by Wen et al. [174]. A global colour shift was also performed to make renderings more visually appealing.

Instead of painterly rendering the segmented regions, researchers also used binary regions to make art. Xu et al. [182], thresholded images to produce Chinese-style papercuts. Later, the same authors completely emitted colour information and studied how intelligent thresholding could lead to stylised black and white images [181], a problem also tackled by Mould et al. [103].

Orchard et al. [110] presented a system that renders images as mosaics of small

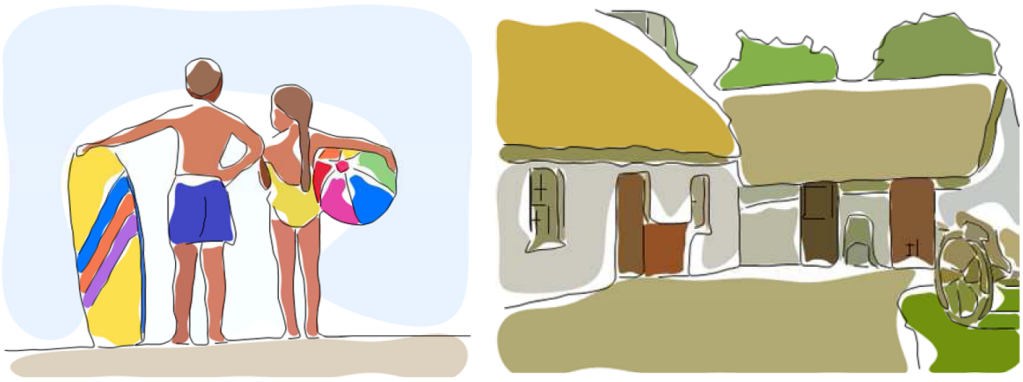


Figure 2-25: Two example colour sketches made using the system of Wen et al. [174].

region cutouts, obtained from a set of training images. Instead of using tiles of regular shapes, such as rectangles, their system is able to tackle cut-outs of arbitrary shapes, hence producing higher quality mosaics. Like others, Orchard et al. [110] also depended on being able to match between source and target tiles. In order to gain computational speed, Fast Fourier Transform (FFT) was used in both matching and colour adjustment.

Chen et al. [16] proposed an interactive system that is able to render human portraits into manga-style line drawings. Their system is examples-based, i.e., a new rendering is made from previously seem examples of the same class. More specially, a database containing pairs of image parts and their corresponding line drawings were trained in advance, by artists. When a new image of a face is presented, the database is searched to yield the best match for each part of the face which are then composited to produce a final rendering. The authors also proposed a separate hair system to increase the aesthetics of the drawings. Qu et al. [119] was able to colourise manga line drawings using an interactive system, where the users only got to put colour scribbles on top.

Bousseau et al. [7] realised that real watercolour paintings are often abstract in nature, so that local stroke rendering techniques might not be a good option. They too relied on obtaining a good segmentation first. Each segment was then abstracted in terms of its shape, colour and illumination. In this way, an abstract representation of the original image was obtained, which can be rendered into watercolour effects.

In the search for making abstract arts, all of the above artistic rendering systems use image segments as basic primitives. There are also other works that create

abstract art using artistic projections [10, 57]. For instance, Hall et al. [57] developed the concept of Rational Tensor Camera, "RTCams", which lifted the traditional constraint of single perspective found on previous NPRP systems and so contribute to NPRP by enabling multi-perspective projection. "RTCams" can be used alone, or compounded to produce more complicated visual effects.

There were also works that uses local image structures to abstract, typical examples include the work of Orzan et al. [111] and Kyprianidis and Dollner [77]. For example, Orzan et al. [111] developed a method to identify salient image structures, which use gradient edges as the basic structural element. It is important to note here that structures in their case meant "non-accidental" features within images and such features were extracted using means of the traditional Gaussian scale space theory. Given the scale space, structures meant meaningful and salient edges according to their importance in the scale space. The intermediate outputs of their system are edge representations of images, each containing a set of salient edges; such representation can then be used as the basis for further NPRP processing. The work of Kyprianidis and Dollner [77] works in a similar fashion to that of Orzan et al. [111], in that they also aimed to find salient edge maps that conforms to local image structures, but using means of bilateral filters.

Evidences on the trend towards abstraction can also be found in the field of NPR from videos [170, 23, 177]. Wang et al. [170] and Collomosse and Hall [23] each independently presented an interactive system to render videos into cartoon-like movies. Just like image-based NPR, both of their works use image segments as basis to support abstract styles. Later, Winnemoller et al. [177] used bilateral filters to enable real-time abstraction of videos. Recently, Fiore et al. [43] presented an interactive painterly rendering system that can be used to create highly stylised animations. The authors introduced the notion of a hierarchical display model (HDM), which is essentially a layered set of 2D brush strokes; HDMs can also be collectively used to enable in-betweening within animations. A comprehensive set of rendering styles were incorporated into their system, such as airbrush, watercolour, ink-wash and etc, all of which made the highly stylised effects possible.

In summary, we have observed how the field of NPRP has moved from modelling brushes and simulating artistic media, to fully automatic rendering system and in the direction of making abstract art. Increasingly higher-level information was

needed to facilitate such evolution. Early works found local image features such as edge maps to be useful, for example when placing strokes; yet, the limitations of low-level image information soon became clear when researchers were trying to extend the gamut of NPRP styles, especially towards the trend in abstraction. Image segments were intensively used to create more abstract art, but manual interaction were often needed to rectify errors and accommodate personal preferences.

There is clear scope for introducing more abstract NPRP styles to the literature, especially styles that are found in the spirit of Kandinsky, Mattise and Mirö. These master artists often use simple geometric shapes and toplogical object structures in their artworks. In order to synthesise such type of art, we would need to bring a much higher level of abstraction to images, such as representing objects as their structural parts and abstracting image segments as pure geometric shapes. It is clear that previously reviewed NPRP systems fall short in such situations; a gap that we seek to fill in this thesis.

Part II

Hierarchical Image Descriptions and Object Structures

Chapter 3

Stable Image Descriptions Using Gestalt Principles

This chapter offers a detailed explanation of the first of the two hierarchical image descriptions proposed in this thesis. This particular image description is based on perceptual organisation and emphasises stable and simple groupings of image primitives, which is commonly referred as "Pragnanz" in Gestalt terms. Here, an image primitive means low-level image features such as pixels and edges; a grouping process then collects such primitives into larger and more meaningful groups. It is commonly argued in the perceptual organisation literature that such groupings often correspond to objects and their structural parts [73, 74], a concept also used by researchers to recognise objects [86, 87]. We allow for more than one such groupings, each of which forms a level of the hierarchical image description. Each level of the image hierarchy conforms to "Prägnanz", which is defined mathematically for the first time. A salience measure is introduced in turn to filter branches in the hierarchy, leaving only the most salient ones. Finally, a novel experiment was conducted to evaluate the image description, both qualitatively and quantitatively.

3.1 Perceptual Grouping: A Review

In the early 1920s, psychologists proposed that *Gestalt principles* play an important role in human perceptual organisation. Initially these consisted of *proximity*,

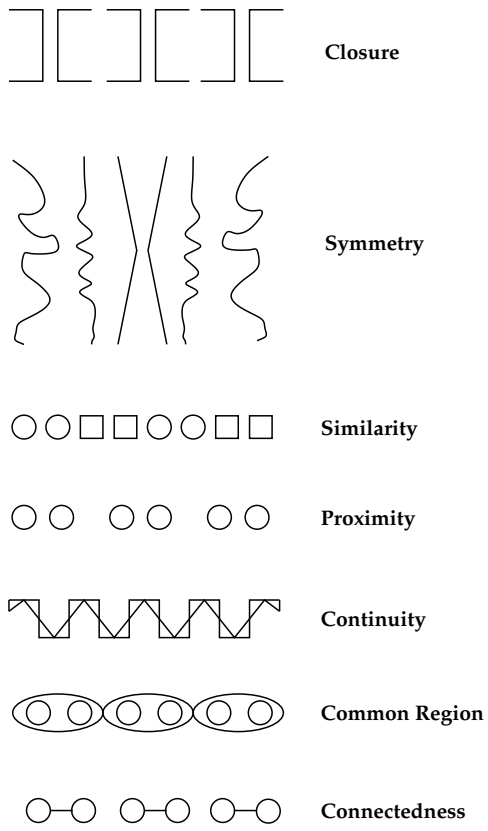


Figure 3-1: Gestalt laws of grouping

continuity, *similarity*, *closure*, and *symmetry*; later *common-region* and *connectedness* were added [73, 74, 175]. Figure 3-1 illustrates the set of seven proposed Gestalt laws. We call these "simple" principles because they act on a few primitives at any one time. Many of these principles have been used in the computational literature. Lowe [87] uses proximity; Carreira et al. use parallelism [14]. Others have sought to use more than one principle at once. Dolan and Weiss [32] use proximity and continuity, a pairing also used by Parent and Zucker [113], and Feldman [38]. Elder and James [34] aimed for contour completion by studying the mutual relationships amongst proximity, continuity and similarity in the task of contour grouping, and concluded that proximity is the most important among those studies. Despite this work, a full computational account of how Gestalt principles interact is yet to be given.

The single idea underlying all of these Gestalt principles is that ordered patterns are unlikely to occur by chance alone. The notion of non-randomness influenced Marr [91], and was explicitly proposed as "common cause" (amongst other names) independently by Witkin and Tenenbaum [178], and Lowe [87]. Lowe argued that

it is highly unlikely for organised image structures to occur by chance; hence they are salient. Yet, common cause can be used without reference to any principle of global organisation, for example a local curve may be the common cause of a set of points, but is not a global constraint.

Common cause is an attractive idea, but is not sufficient. Consider a simple example in which two people face one another. We might be tempted to group them, describing them as a single entity (as we have above). If now we "zoom out" to reveal that each person is at the front of a queue, each facing the other we may decide to separate them, grouping them instead with the queue to which they belong. More generally, context — by which we mean the presence (or absence) of structures in an image — affects the outcome of grouping. Therefore, we argue, some notion of global organisation must be included in any account that seeks to form groupings, and such an account should be integral to the way in which Gestalt principles are combined. Some of the above work operates hierarchically [87, 32, 38] and can deal with more than one primitive at a time. Yet it remains true that very few of them make use of any principle of global organisation.

Prägnanz is the Gestalt principle that seeks to organise primitives in a "global" sense. Introduced by Wertheimer [175], Prägnanz was developed by Koffka [73] who advocated that *"of several geometrically possible organisations that one will actually occur which possesses the best, simplest and most stable shape."* Kanizsa [71] too suggested Prägnanz implies an orderly, rule-based , non-random and stable organisation of primitives.

Computational techniques that aim to find best groupings in a global sense do exist. Guy and Medioni [54] combine proximity and continuation to find contours, which differs from those schemes previously cited in that a global voting scheme was introduced, where each pixel gets votes from all other ones. Sarkar and Soundararajan [132] used a modified version of normalised cut, and applied it to the adjacency graphs obtained using a small set of Bayesian Networks, each of which corresponds to a certain Gestalt principle and is trained by a set of mutually competing automatas.

Despite these successes, they do not provide an explicit definition of Prägnanz. Of the literature we read, only Ommer and Buhmann [107] explicitly define Prägnanz, in their case as the minimum of an entropy function based on the

probability that pairs of primitives should be grouped. By minimising entropy, these authors ensure simple groupings, as do the other global methods. But Simplicity is just one half of Koffka's requirement that global organisation should be simple, but there is no guarantee that the organisation is stable.

3.2 Overview

Our under-pinning strategy is analogous to building a jig-saw by assembling small pieces of a puzzle into ever larger pieces that are more efficient to work with. Each stage of aggregation changes the way the jig-saw pieces are characterised, moving from pixels through to a stable, hierarchical description of the image. Since each stage operates with primitives of different characters it is natural that each stage employs its own measure to control its aggregation process. Nonetheless each measure has been subject to the common constraints that (i) it produces a *plausible grouping* and (ii) is easy to compute. Our approach is empirically justifiable; results in Section 3.8 compare favourably to groupings that humans make.

Our approach has three main stages. The first stage aggregates pixels into line elements and region elements using methods available in the general literature (Section 3.3). The second step uses proximity and common region to identify stable groupings of image primitives, which in our case are lines, see Section 3.4. The third and final stage arranges these groupings into a lattice, which is then examined to determine salient groups, as explained in Section 3.5. We call such groups maximally stable salient groups (MSSG). Here, maximally stable means minimal change with respect to a vector of control variables; salient means the most significant of all the maximally stable groups. The lattice also provides us with the notion of "grouping scale". A diagram illustrating the whole system is shown in Figure 3-2.

In the reminder of this chapter, we describe how the hierarchical image description is constructed, including our definition of Prägnanz in Section 3.4 and the introduction of the lattice structure and its usage in Section 3.5. In Section 3.8, we first provide a detailed description of the experiment that was conducted, followed by the experiment results. Finally, a conclusion and discussion on future work is offered in Section 3.9.

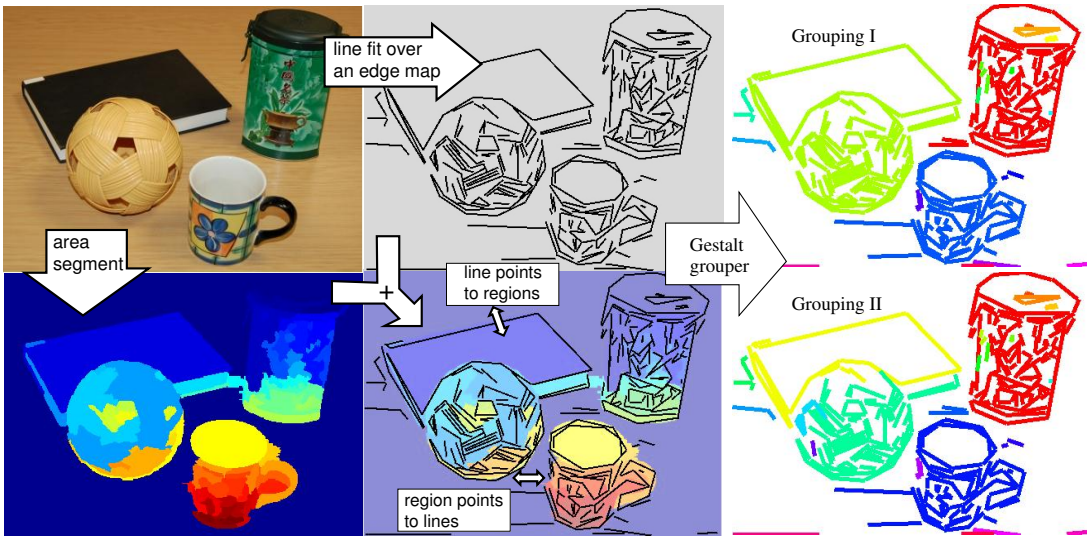


Figure 3-2: A picture is processed into edges and areas. Lines are fitted to edges which reference back to areas. Our Gestalt grouper optimally connects lines into groups. In this and all Figures in this chapter, groups are colour coded; singleton groups are in grey. Please refer to the electronic version of the thesis for best viewing results.

3.3 Deciding Grouping Primitives

The input to our grouping algorithm is a single image made of pixels. Pixels are grouped during the first stage into straight lines and homogeneous regions. Straight line primitives are fitted to an edge map obtained by thresholding an Elder-Zucker edge detector [35], which requires monochrome images. Region primitives are connected pixel sets, determined by a colour-image segmentation from Sinclair [148]. Both of these algorithms claim some grounding in perception, as will be explained below. However, it should be noted that our proposed grouper is expected to work with other edge detectors such as Canny and image segmentation techniques such as Mean Shift. For reasons that will become clear later in this section, we only require a degree of overlap between gray-scale edges and colour segments.

The Elder-Zucker edge detector has been specifically designed to match human low-level visual processing [35]. This choice is justified by the fact that a local scale control was used, hence they claim that the results of this particular algorithm correspond more closely to the edge map that human beings perceive. Furthermore, as previously mentioned, Elder [33] demonstrated in an immediately following paper that it is possible to reconstruct grey-scale images from

Elder-Zucker edge maps given blur and scale of individual edges, by solving a Laplacian globally. This image reconstruction technique has been fully implemented. However, it was found that it performed badly with textured images. Nevertheless, Elder showed us that an image reconstruction algorithm from edge maps are possible, which could be an extension to this study.

The image segmentation technique used is developed by Sinclair [148] from AT&T labs, Cambridge. The important aspects of this particular technique are that at first, it is originally developed for semantic image retrieval systems hence fits well with the semantics of perceptual grouping. Secondly, it employs a colour edge detector which emphasis more on the colour than intensity changes, hence, it ensures that the grey-scale edges are independent of the area map. Sinclair's segmentation algorithm emphasises on the photometric properties of images, and is also claimed to match better to that of human visual systems than other segmentation techniques. Moreover, it also has been successfully applied in content-based image retrieval tasks [163].

Since it is easier to merge rather than split groups, we have tuned both the line fitter and region segmenter to over-fit the data: fitting produces many straight lines and the image is typically over segmented. Nonetheless, pixels are aggregated into useful primitives: lines and regions. A bipartite graph $G = \{L \cup R, A\}$ is constructed from the lines L and regions R; $A \subset (L \times R)$. A line is connected to those regions that it covers. Conversely, each region is connected to those lines that cover it. This graph is conveniently mid-level in the sense that its primitives are super-pixels, but is nonetheless a weak description.

Figure 3-2 illustrates the proposed scheme with the image called "desktop", which will be used as a running example throughout the chapter. It shows with primitives output by the standard edge detector and region segmenter. The next step is to group line primitives using Gestalt principles.

3.4 Forming Stable and Simple Groupings

The next stage of our approach uses the bipartite graph as input. Here it is important to distinguish between a *group*, which is a collection of image primitives, and a *grouping* which is a collection of distinct groups covering an image.

We form groupings from line segments using the simple Gestalt principles of proximity and common region. We use a vector of variables to control the grouping process, and produce many groupings by varying element values in the control vector. The quality of any grouping is measured by its stability and simplicity; this is Prägnanz as required by Koffka. We select the groupings that are both stable and simple. Our more detailed account opens with an explanation of the elementary grouping process. We then define Prägnanz, using our definition to bring the elementary process under control and so produce stable, simple groupings.

We first explain how we combine simple Gestalt principles via logical statements to make groupings, then how we order that process using Prägnanz.

3.4.1 Incorporating Gestalt Laws into the Grouper

We group line primitives by combining two simple Gestalt principles: proximity and common-region. We adopt a simple approach that is strongly influenced by the work of Feldman [38], who advocates combining simple Gestalt principles via logical propositions.

Accordingly, we define the proximal distance $d(.)$ between any pair of line primitives (i, j), to be the smallest Euclidean distance between their end points. For line segments we define, giving a "proximity proposition" that depends on a threshold x_1:

$$p(i, j|x_1) = \begin{cases} 1 & \text{if } d(i, j) < x_1 \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Similarly, we define a "common-region proposition":

$$c(i, j|x_2) = \begin{cases} 1 & \text{if } r(i, j) > x_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

in which $r(i, j)$ counts the number of regions the line primitives have in common and x_2 is another threshold. The number of common regions is readily determined by simply intersecting the list of region identifiers associated with line segment.

A simple "or" combination was found to be an effective way to combine them

$$a(i,j|[x_1,x_2]) = p(i,j|x_1) \vee c(i,j|x_2) \tag{3.3}$$

If $a(i,j|[x_1,x_2]) = 1$, then the line primitives are connected; they are disconnected otherwise. The value of threshold vector $\mathbf{x} = [x_1, x_2]$ therefore determines an adjacency matrix. Other means of combining Gestalt principles via logical operations were tested as well, $a(i,j|[x_1,x_2]) = p(i,j|x_1) \wedge c(i,j|x_2)$ being one of them. This is telling the grouper to group a pair of line primitives, if they are both proximal and cover a few common regions. We found such combination yields too few groupings, mainly due to the restrictiveness of the the $\wedge$ operation. The lack of potential groupings then made the later analysis that explores the space of all groupings badly conditioned. We have also experimented with adding in more Gestalt principles, such as continuity and parallelism, with various logical combinations; but found the space of groupings produced either too restrictive, or noisy. It is worth mentioning that the problem of how Gestalt principles work in a collective fashion still remains unknown in the field of psychology [34].

Because connectedness is a transitive relation, an adjacency matrix will partition its nodes into groups, which are connected sets. The collection of all groups in a given adjacency matrix defines a grouping that partitions the image primitives. If $(i,j) \in g$ and $(i,k) \in g$, then $(j,k) \in g$. Such groups can be identified with any other of their elements and so can be written as an equivalence class $g[i|x]$:

$$g[i|\mathbf{x}] = \{j : \exists j_k : (j,j_1) \in g \ldots (j_n,i) \in g\} \tag{3.4}$$

$$Q(\mathbf{x}) = \cup_i g[i|\mathbf{x}] \tag{3.5}$$

The groupings we obtain clearly depend on the values of control vector. The choice on the set of values remains unknown, and such choice in general depends on the context. The role of Prägnanz is to select values of the control vector that gives a stable, simple partition of Q; which we write as $Q(\mathbf{x})$ so as to emphasise the control that the threshold vector has.

3.4.2 Pragnanz: Locating Stable and Simple Groupings

The stability of a description is a very important idea in Computer Vision and is one we use here. Stability is excellently exploited by Maximally Stable Extremal Regions [93]. MSER changes a scalar threshold over a grey level image, looking for binary regions of stable area. We act similarly but: (1) use a vector of thresholds, not a scalar; and (2) we look for groups with a stable entropy-like function.

Our definition of Prägnanz does not depend on the details of the way in which a partition (adjacency matrix) is formed. Potentially, it could be used as a controlling principle for any grouping algorithm whose output depends upon a set of control variables, not just our own grouper.

Let $\mathcal{P}$ be a set of image primitives, straight line segments in our case. Let $Q(\mathbf{x})$ be a partition of $\mathcal{P}$ that depends on a vector of control variables $[x_1, x_2]$, $\mathbf{x} \in \Re^n$. In our case this is the vector of threshold values used in the "proximity" and "common-region" propositions of Equations(3.1) and (3.2). We suppose that $f(Q(\mathbf{x}))$ is a scalar valued function that measures some property of a given partition such as information content.

We define a partition to be *(consistent with Koffka's) Prägnanz*, if it is stable and can be justified by appeal to simple Gestalt principles. We define the *stability* of a partition as the magnitude of the gradient of $f(.)$ with respect to the control vector:

$$s(Q(\mathbf{x})) = \left(\sum_{i=1}^{k} \left(\frac{\partial f(Q(\mathbf{x}))}{\partial x_i} \right)^2 \right)^{1/2} \tag{3.6}$$

We define a partition to be *stable* if $s(Q(\mathbf{x})) = 0$. In practice, the discrete nature of the control variables means zeros are rarely observed. Instead we seek local minima at the bottom of watershed regions.

As mentioned, we wish $f(.)$ to measure the information content of a given partition. Here we assume only the adjacency matrix A for a given control vector; this is acceptable since an adjacency matrix can be partitioned into connected sets, each set being a group. We define $f(.)$ as

$$f(Q(\mathbf{x})) = -G(\mathbf{x}) \log \frac{A(\mathbf{x})}{G(\mathbf{x})} \tag{3.7}$$

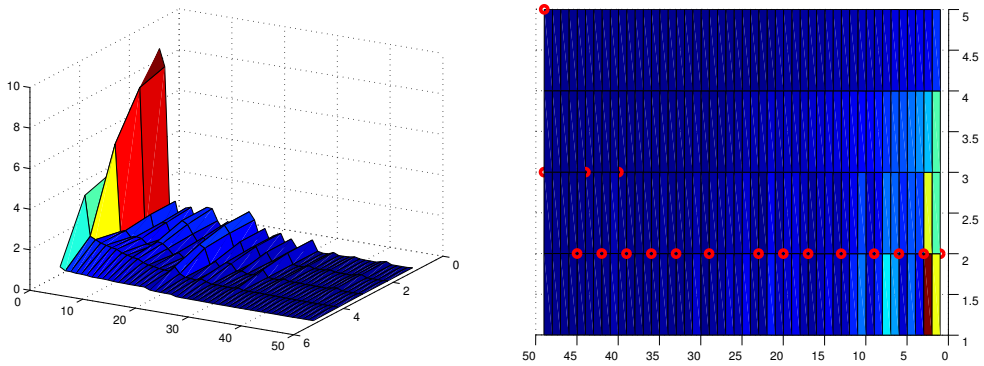


Figure 3-3: Left: a surface plot of the function $s(Q(\mathbf{x}))$; right: local minimas found at the bottom of watershed regions of $s(Q(\mathbf{x}))$, which correspond to stable groupings.

In which $G(\mathbf{x}) = N_g/M_g$ is the normalised number of groups, and $A(\mathbf{x}) = N_a/M_a$ is the normalised number of arcs; N_g and N_a be the number of groups and arcs in a particular partition, respectively, whereas M_g is the maximum number of groups in any partition, and M_a is the maximum number of arcs in any partition.

Our definition of $f(.)$ is related to the number of binary digits required to encode the grouping, when compared with the most complex alternative. $A(.)/G(.)$ is the average number of arcs in each group, so $Glog(A/G)$ is a crude measure of information content. We appreciate this function is somewhat ad-hoc, and no doubt there is ample room to investigate alternatives. But our measure has several features in its favour: (i) it depends only on the adjacency matrix, so it can be used with any grouping process; (ii) it is quick and easy to compute; (iii) it turned out to be most reliable measure from amongst all those we tested. A plot of the differentiated surface of our $f(.)$, i.e., $s(Q(\mathbf{x}))$, is shown in Figure 3-3(a); Figure 3-3(b) offers a flat view of the same surface, with points corresponding to stable groupings highlighted.

There is usually more than one stable value of the control vectors, hence more than one Prägnanz partition of an image. Figure 3-4 shows four stable groupings of our example image, "desktop", ordered by simplicity — that is, the value of $f(.)$. We make several important observations: (i) We have managed to "pull" objects (or the majority of their parts) from the image. (ii) Identifiable objects appear in different images, that is with different values of $f(.)$; the ball and book are separated in one image but not others, for example. (iii) There is a sense of

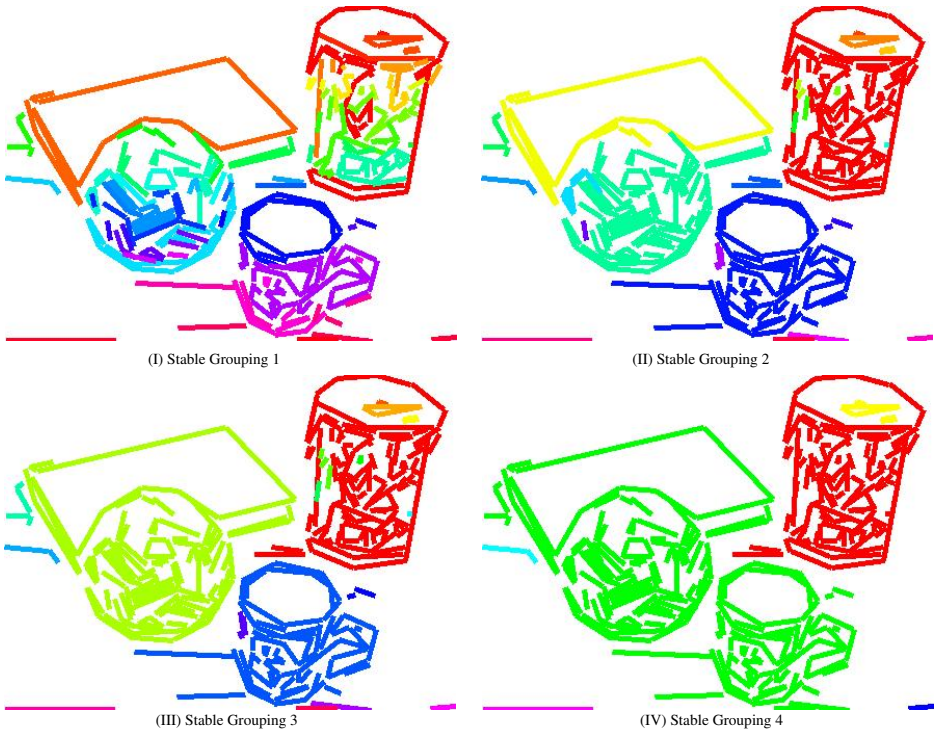


Figure 3-4: Four stable groupings ordered in terms of their simplicity. In this and all Figures in this chapter, groups are colour coded; singleton groups are in grey. Please refer to the electronic version for best viewing results.

"scale", as f rises so groups get larger in terms of the number of primitives they contain, and their total number falls. We call this the *grouping scale*. (iv) Some objects appear to persist over scales.

A qualitative description can now be given. At the "finest" scale, objects such as the ball, mug and tea-box tend to be rather over-segmented, for example. However, the group that corresponds to the book stays unique and groups from the mug have a clear top, middle and bottom arrangement. Grouping II is of "middle" scale, at which there is a clear separation of the four objects, viz, book, ball, tea-box and mug. The ball and the partially occluded book were merged into a single object in Grouping III. Grouping IV is at the "coarse" scale which tends to discriminate yet larger objects, in this case, the book, ball and mug were grouped into one, while the tea-box persist as a single group. If we made the scale large enough, then all primitives would be grouping into one.

The purpose of the final stage of our approach is to locate the most salient groupings from amongst the universe of groups, which is generated by the union of all Prägnanz partitions. The salient groups are the final output of our method.

3.5 Locating Salient Groupings

Having identified a set of stable groups using our definition of Prägnanz, we set out to filter out a subset of salient ones, so to complete the definition of maximally stable salient groups. From Figure 3-4 we observe that salient groups appear in many stable groupings, rather like salient edges persist over scale [179]. In our case we exploit the hierarchical (set-subset) relationship between groups to define *levels*, each comprising of disconnected groups.

The universe of all stable, simple groups naturally forms a lattice structure via the subset relation. The output of the Prägnanz grouper is a level in the lattice. The lowest level of all is the set of all line primitives; the highest level is their union. Levels lower down the lattice contain many small groups, those higher up contain fewer, larger groups. Hence, the levels of the lattice can be informally thought of as a kind of "grouping scale".

We want to pick salient groups out of the lattice. Following Witkin's spacial filtering over edges [179], we define the *longevity* of a group to be the number of levels that it persists, with the hope that groups with large longevity are salient. However, the problems of doing just that are two-folds. At first, groups coming from image noise tend to stay over scale, hence having large longevity. Secondly, under one of the main assumptions of our system, there is no single grouping that is favourable, hence the salience measure should be dependent upon user preference. In order to tackle both problems, a single parameter N indicating how many groups that an user wants is introduced, i.e., the scale to which an user interprets the image. N can also be viewed as prior information for an image.

In order to tackle the short-comings of longevity, we introduced two other properties to filer out salient groups. Therefore, our measure for the salience of a group has three heuristic components:

- Just as salient edges persist over scale [179], so do salient groups. The *longevity*, λ, of a group is the number of lattice levels on which that group appears. If H is the height of the lattice, then λ/H is the normalised longevity.

- Salient groups will tend to be of middling scale, groups that are too small tend to be the result of over fitting, groups that are too large tend to result

from under-fitting. The *size*, σ of a group is the number of line primitives in the group. If we expect N groups to be in the final output, and there are L line primitives, then $\sigma_0 = L/N$ is the expected size of a group. Hence σ_0/σ is the expected size relative to the group size, and $|1 - \sigma_0/\sigma|$ is departure of the group's relative size, which we use as the "size salience".

- The visual *area* of a group is another factor in determining salience. We define the area, α, of a group to be the number of primitive regions associated with the group (recall the fact that every line primitive "points to" the regions it covers). As with size, we prefer groups of middling area relative to the expected area $\alpha_0 = |R|/N$, in which $|R|$ is the number of region primitives, hence we use $|1 - \alpha_0/\alpha|$ as the "area salience"

We note we have defined size and area in such a way as to be independent of physical scale (measured in pixels, say).

We define the *salience* of a group as

$$\rho(N) = -\frac{H}{\lambda}\left|1 - \frac{\sigma_0(N)}{\sigma}\right|\left|1 - \frac{\alpha_0(N)}{\alpha}\right| \tag{3.8}$$

in which we have explicitly highlighted the dependency of this measure on a free variable, which is the number of groups N. The value of N can be fixed by a user. Once N is fixed, a salience score is computed for each group in the lattice of groupings and the N most salient groups are picked. As previously mentioned, our saliency measure favours medium-sized groups in terms of number of line segments and associated areas and grows negatively when *longevity*, λ, increases.

This process picks only a subset of line primitives — those that belong to the N groups. If a full partition is desired by iteration, the remaining primitives can be distributed amongst those picked by associating the un-picked group closest to any picked group, thereby reducing the set of un-picked primitives and growing one of the salient groups; the iteration ceases when no primitives remain to be associated.

Following the "desktop" example, Figure 3-5 and 3-6 provides two set of results. Each corresponds to a different N parameter, which specifies the number of salient groups wanted. In Figure 3-5, 3 salient groups were pulled out from the lattice (left), based on which a final overall grouping was produced (right). As can be

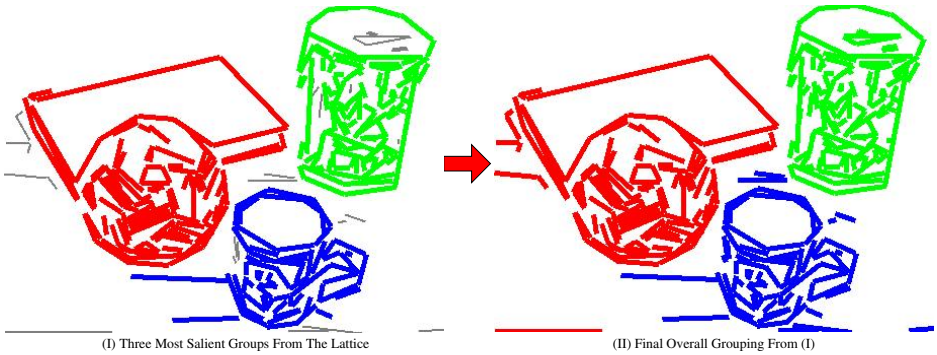


Figure 3-5: Left: three salient groups pulled from the lattice; right: overall grouping based on salient groups shown on the left

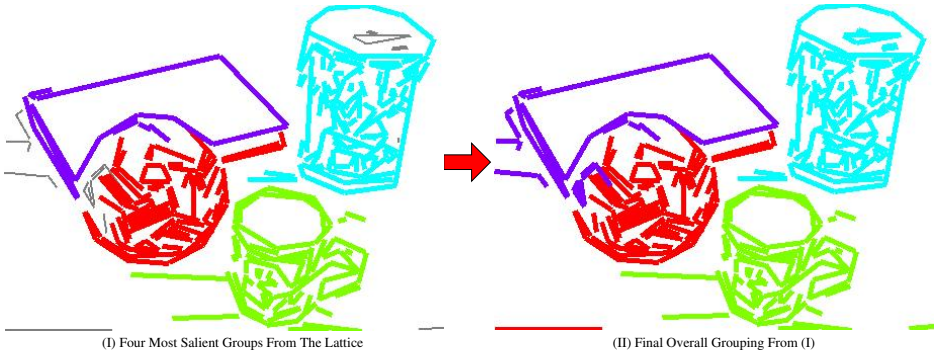


Figure 3-6: Left: four salient groups pulled from the lattice; right: overall grouping based on salient groups shown on the left

seen, the salient groups chosen are of coarse grouping scale, in that groups tend to correspond to larger entities. In this particular example, the ball and book were pulled out as one. Figure 3-6 follows the same pattern as Figure 3-5, but with $N = 4$. Objects of a relatively finer scale were pulled out, which is reflected by the fact that the occluded ball became separated from the book behind. Meanwhile, we note that both the tea-box and mug change little in each image, suggesting they are highly salient.

The salient groups are determined cross different levels of the lattice, and so there is no fixed value of the control vector that will yield such a result. It is as if the control vector were locally adapting to different parts of an image, but doing so with reference to all other parts of the same image. The salient groups are stable and simple. We have conducted experiments to assess their semantic value, which we describe in Section 3.8.

3.6 Relating Edge Primitives to Areas

Having identified maximally stable salient groups (MSSGs) all steps of our algorithms are described. Here we provide the useful addendum of relating regions to lines in MSSGs. Each MSSG partitions the set of line primitives, depending only on the number of groups (equivalence classes) required. We want to partition region primitives correspondingly. However, ambiguity makes it is difficult: each line can point to many regions, and there is no guarantee that any given region points to lines in a single MSSG.

Our solution begins with regions that are identified with a single group. Next the general background is identified as the most shared region; this assumed salient objects are more-or-less evenly distributed over the image rather then nesting, one inside the other. Finally, each remaining region is associated with the group of most similar colour. This technique works well on images of rather plain backgrounds, but becomes unreliable when the background is more textured which results in over-segmentation. We resolve this by constraining the size of the corresponding region of a group by its convex hull. Figure 3-7 shows a result, Section 3.7 provides other examples.

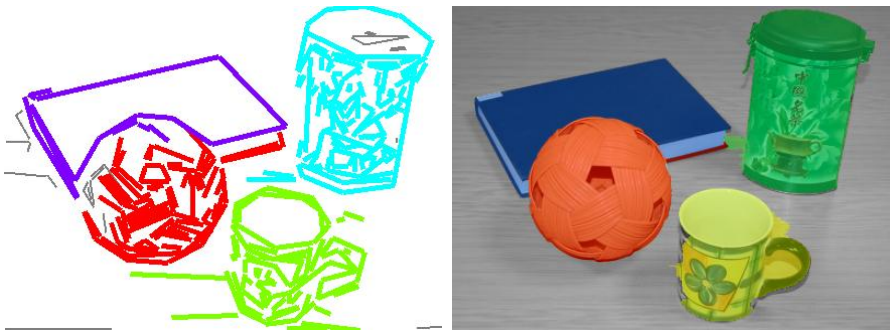


Figure 3-7: Left: four salient groups from the hierarchy, non-salient groups in grey; right: after regions are related to lines (subsection 3.6), tinting shows salient regions, background in grey;

3.7 Qualitative Examples

Before describing our quantitative experiment, a few qualitative examples of MSSGs are presented in this section. For better visualisation, we tint regions we get from Section 3.6.

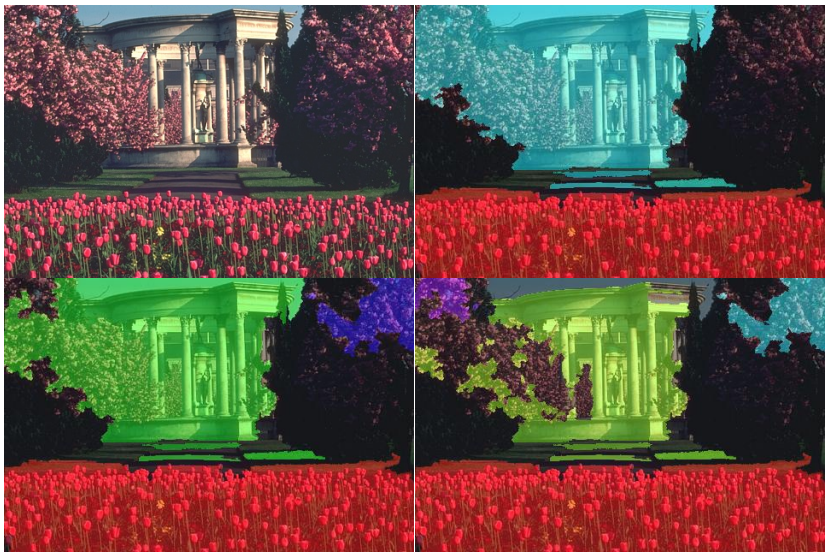


Figure 3-8: Top left: original image; top right, bottom left and bottom right shows MSSGs picked when N is increased from 2 to 4

Figure 3-9: Left: original image; middle and right: MSSGs picked when $N = 2$ and $N = 4$, respectively

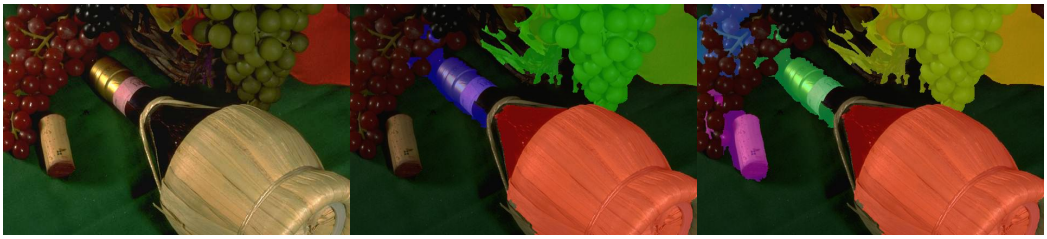


Figure 3-10: Left: original image; middle and right: MSSGs picked when $N = 3$ and $N = 5$, respectively

As can be seen from Figure 3-8, when $N = 2$, the two MSSGs picked correspond to the two most salient objects in the scene, viz, the building as background and the flowers as foreground. As the number of groups increases, the foreground stays unchanged, but the background separates; at $N = 4$ we see left blossom, right blossom, building and flowers.

A similar trend continues with results shown in Figure 3-9. In Figure 3-9, there is a background and foreground separation when $N = 2$, whereas, setting $N = 4$ breaks the foreground into meaningful parts, viz, the boat, the beach, and stairs. Figure 3-10, shows that new objects can be included when all objects are of about the same visual scale. Here, as N rises, more salient objects are added in.

The trend shown in these examples is clear: as the number of groups rises, the different parts of each picture become resolved as the more weakly connected groups begin to split. On occasion, when objects are of about equal scale, new groups appear, such as the right blossom in Figure 3-8 and the grapes and cork in Figure 3-10.

3.8 A Quantitative Experiment

In this section we describe an experiment: its method, its results and our conclusions. The experiment sets out to determine the extent to which our groupings are semantically meaningful to humans. We used human based segmentations as a control, and normalised graph-cuts [142] as an alternative popular automatic grouper.

The basic idea of our experiment is akin to the Turing test: to show people groupings, and ask if they agree or not with the statement "this grouping was produced by a human". We define the "semantic inefficacy" of any grouping process to be the number of disagreements. This basic idea needs a little modification as we expect degrees of disagreement rather than a binary decision: people may agree a grouping is "about right", or "acceptable in parts", for example. We therefore instructed our experimental volunteers to *edit the grouping in a picture until it is in a state they believe is plausibly produced by a human* — that is, passes the Turing test for them. The number of edits is a measure of the disagreement, and becomes the basis for our measure of semantic inefficacy.

We prefer this approach to the alternative of comparing our groupings with a fixed set of groupings produced by humans for several reasons.

- Firstly, an obvious but important observation is that humans disagree with one another as to the particular grouping for any given picture. In response, our measure is based just on the number of edits to some target grouping

that a user may never fully make public. This contrasts with using a fixed set of target groupings produced by humans.

- Secondly, and more subtly, a fixed set of groupings may represent only a subset of what humans agree with which may inadvertently introduce a bias. Our experiment mitigates against this because it widens the number of humans taking part, and those that segment the images are not the same people as those that judge the results.

- Thirdly, the human produced groupings are subject to the same test as automatic groupings, rather than being given a privileged position. This is, of course, necessary for our "Turing test" to be valid.

- Finally, our experiment can measure the *disagreement* between humans, which can be used as a basis for comparing automated processes.

We measured the semantic inefficacy of two automatic processes: our own described in this chapter, Shi and Malik's normalised cut [142]. Both of these automatic processes require a single input parameter, which is the number of groups required in the final output. Thus both our method and the automatic alternative depend on a single number input by the user. The control set of human-made groupings were produced independently of us, they come from Berkeley segmentation database [92]. We proceeded as follows.

3.8.1 Experimental Method

Both Berkeley segmentation database and Normalised cut deliver area based groupings. To accommodate with the line segments groupings we produce, the same set of line segments are overlaid on top of segmentations returned by Berkeley and Normalised Cut, and those lie in the same region form a group. In this way the user was presented with data consistent with our grouper. The alternative of producing area segmentations from our grouper requires an additional step that ascribes "foreground" and "background" to each group. Such a step is necessarily heuristic (as many optical illusions show).

First we decided on a set of six photographs to be used in our experiment, see Figure 3-11. The only constraint was that the photographs must be present in

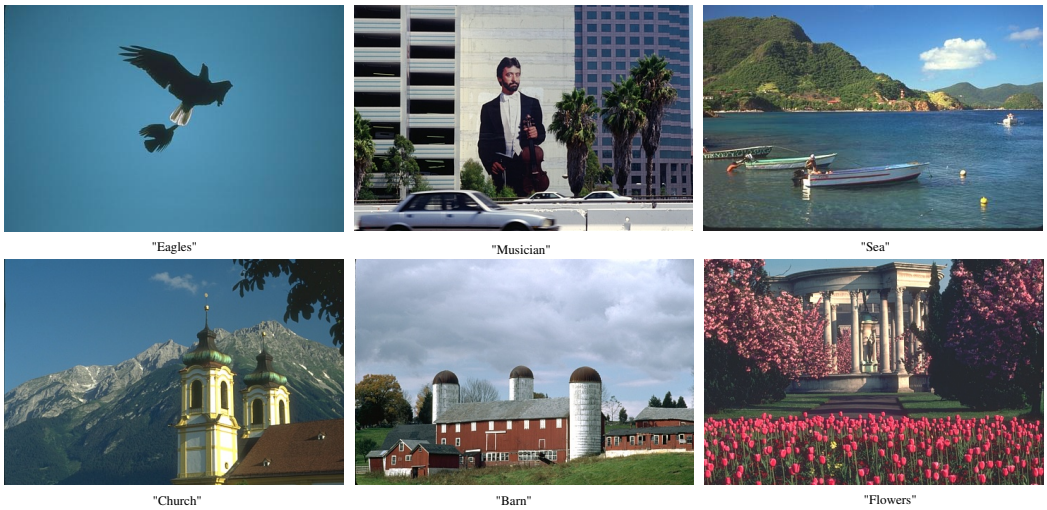


Figure 3-11: Six images from Berkeley Segmentation Database used in the experiment. Top: (left) "Eagles", (middle) "Musician", (right) "Sea"; bottom: (left) "Church", (middle) "Barn", (right) "Flowers"

Table 3.1: Experiment data description

Image Name	Number of Groupings	Number of Groups
"Eagles"	2	2 and 3
"Musician"	2	9 and 14
"Sea"	1	12
"Church"	1	6
"Barn"	1	9
"Flowers"	1	5

Berkeley's database, so that more than one human grouping was available for each photograph. Each photograph was grouped by each automatic process, configured to produce more than one grouping result. Specifically, the automatic grouper was configured to produce the same number of groups as present in the control set from Berkeley. For example, suppose a photograph has two entries in the control set, one with 3 groups another with 4; then the automatic processes were configured to produce one solution with 3 groups and another with 4. We acknowledge there may be acceptable solutions outside this range, but we choose to restrict ourselves to the number of groups present in the existing control set. The number of salient groupings chosen for each image and how many groups each contains can be found in Table 3.1.

Thus we generated a collection of photographs, each grouped by humans and machine, each grouping process produced more than one solution. We continued

by showing an original photograph and one of its groupings to a human, who then edits the grouping to their satisfaction. The subject is never told there is more than one grouper that produced the results they see, and in particular never told that some have been produced by other people. The edits a subject makes are recorded by software that controls the experiment, including allowing simple editing: "split this group" and "merge these groups" being the fundamental editing operation.

We used 10 human subjects. The photographs were presented to each subject in a random order known neither to the subject nor the experimenter but chosen by software. Randomising the order of photographs has several benefits. Subjects may tire of editing, and perhaps become less particular as time progresses - randomising normalises over this effect. Also subjects may "learn" that a particular photograph has a particular groupings; randomising normalises over this variable too.

3.8.2 Experimental Results

This section presents results obtained from the three grouping methods we used: the human control, Shi and Malik's normalised cut, and our hierarchical, Prägnanz approach. Normalised cut is, of course, an excellent general purpose algorithm with uses beyond Gestalt grouping. Nonetheless it remains a popular approach and the basis of several grouping algorithms with aims similar to our own. Additionally, it requires just one input parameter. It therefore makes a good choice of algorithm against which to compare our own, special-purpose approach.

For each of the three grouping techniques (human, normalised-cut, our method), there are five representative groupings used in the experiment. Table 3.1 shows the number of groupings selected for each image, together with their sizes. All 24 groupings used in the experiment are in-turn provided in Figure 3-12, where each column corresponds to a particular grouping technique and each row has a common number of groups.

It was the colour-code grouping images in Figure 3-12 that our experimental subjects were allowed to edit. The distance between the subject's edit and the original grouped image was used as a measure of "semantic inefficacy": a greater distance is taken to mean a less efficacious grouping method. To compare the

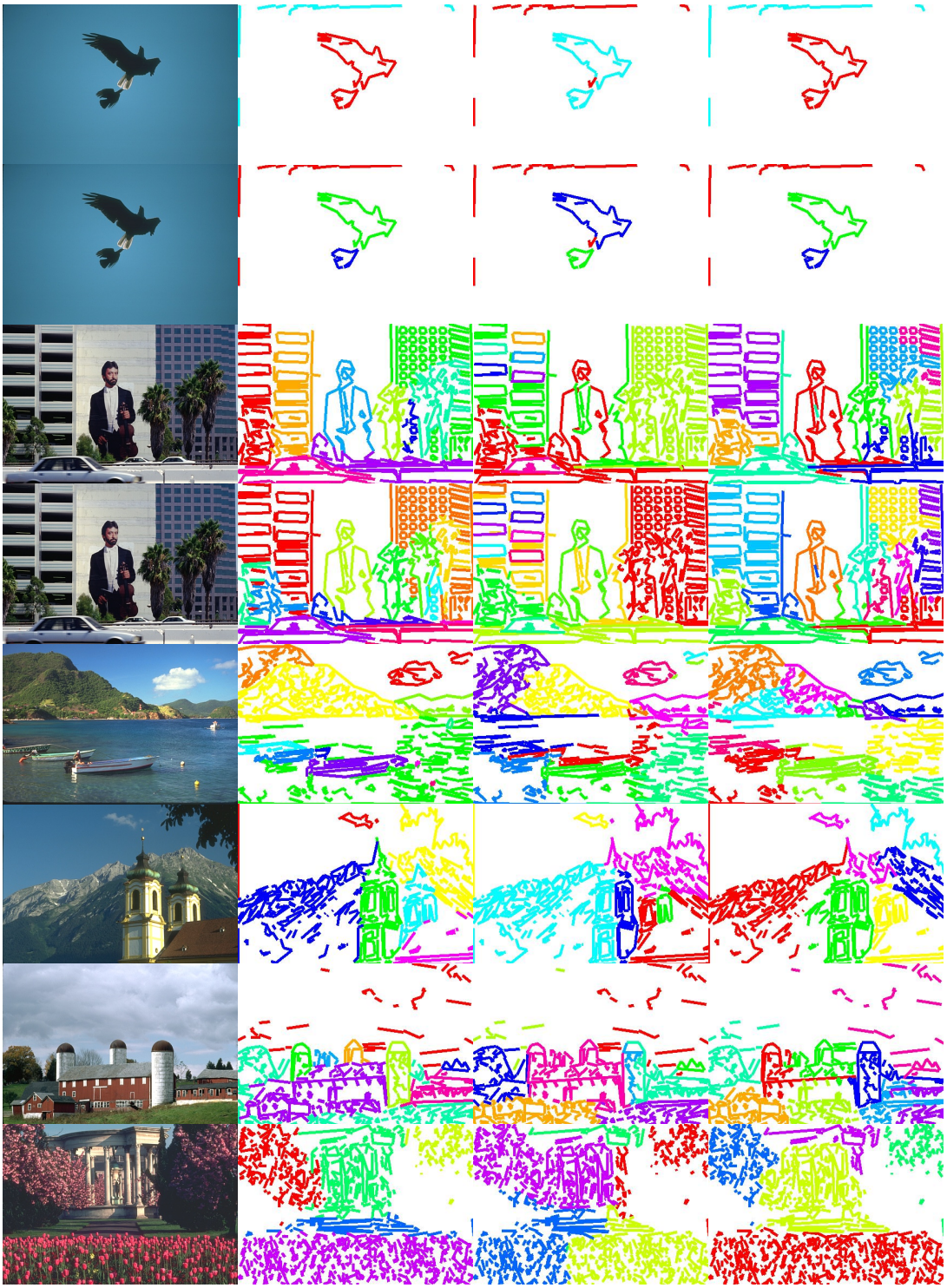


Figure 3-12: All 15 groupings used in the experiment. Columns left to right are: original image, human Control, Normalised Cut, our method. Each row is an unique way of forming groups in an image.

results we used two different graph similarity measures, so as to guard against possible bias in any one of them. One of the measures uses Laplacian eigenvalues [17], the other is a graph edit distance [122]; we briefly explain both graph theoretic measures below, however, the details of which can be found in the original papers.

The first method is based on the spectral graph theory [17]. For two graphs A_1 and A_2, where A_i is the adjacency matrix for the graph G_i. We first compute the Laplacian matrix for the two graphs, $L = D - A$, where D is the degree matrix, whose elements are given by $D(u,u) = \sum_{v \in V} A(v,v)$. From the Laplacian matrix we perform eigen decomposition, $L = \Phi \Lambda \Phi^T$, where $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_{|V|})$ is the diagonal matrix with the ordered eigenvalues as elements. Since the two graphs to be compared contain the same number of nodes in our experiments, we may extract the eigenvalues from matrix Λ for each graph to construct a same length feature vector, $B^T = (\lambda_1, \lambda_2, ..., \lambda_{|V|})$, where $|V|$ is the node number for two graphs. We can then use the Euclidean distance between two feature vectors $D(B_1, B_2)$ as the similarity value between two graphs G_1 and G_2.

The second method to measure similarity between two graphs is based on graph edit distance [122]. We first sort the nodes of a graph into a string. The edit distance between two graphs is found by finding the sequence of string edit operations which minimises the cost of the path traversing the edit lattice, which is determined by the components of the leading eigenvectors of the adjacency matrix.

In Tables 3.2 and 3.3 we show the mean similarity values for the three different grouping processes, computed using Laplacian eigenvalue and graph edit distance respectively. The numbers in brackets are standard deviations. Each table has three columns, one per grouping process. Each table has 8 rows, each representing a fixed number of groups, so these tables correspond 1-1 with the images in Figure 3-12. The first two rows represent two distinct interpretations of the eagles photograph seen in Figure 3-11: one is a single group comprising a pair of eagles, the other is two groups of one eagle each. The third and fourth rows corresponds to two unique groupings of the "musician" image; and the rest rows contains three mean similarity measure from a "sea" grouping, a "church" grouping, a "barn" grouping and a "flowers" grouping, respectively.

In Figure 3-13 and 3-14, we show the same data presented in graphical form. Plots

Table 3.2: Similarity measurement for different methods by using Laplacian eigenvalue

	Human Control	Normalised Cuts	Our Result
Eagles I	0 (0)	0.086 (0.046)	0 (0)
Eagles II	0.0205 (0.065)	0.063 (0.022)	0.041 (0.086)
Musician I	0.17 (0.19)	1.12 (0.12)	0.17 (0.22)
Musician II	0.049 (0.053)	1.48 (0.099)	0.22 (0.38)
Sea	0.12 (0.23)	0.61 (0.18)	0.36 (0.40)
Church	0 (0)	0.256 (0.21)	0.085 (0.085)
Barn	0.030 (0.064)	0.14 (0.038)	0.064 (0.077)
Flowers	0 (0.0013)	0.068 (0.014)	0.014 (0.0092)

Table 3.3: Similarity measurement for different methods by using spectra edit distance

	Human Control	Normalised Cuts	Our Result
Eagles I	0 (0)	20.9 (19.50)	0 (0)
Eagles II	7.41 (23.43)	59.4 (20.87)	14.8 (31.24)
Musician I	308.4 (219)	756.9 (83.90)	333.5 (145.62)
Musician II	251.1 (220.44)	757.1 (119.55)	229.7 (163.48)
Sea	111.9 (202.13)	507.1 (165.02)	404.5 (220.65)
Church	0 (0)	222.14 (85.58)	52.43 (30.02)
Barn	285.06 (335.97)	499.48 (124.30)	212.49 (170.78)
Flowers	142.63 (451.06)	1024.70 (282.48)	321.57 (344.01)

in Figure 3-13 show the similarity results computed using Laplacian eigenvalue method, while those in Figure 3-14 show the results by using the graph edit distance. Student T-tests are used to quantify differences between each and every pair of distributions shown in these figures.

Table 3.4 shows the student T-test results on Gaussians derived from Laplacian graph similarities, whereas results corresponding to Graph Edit Distance similarities are provided in Table 3.5. A low absolute value of the T-test implies a closer distance between the distributions. The sign of the T-test locates the mean of the test distribution with respect to the reference distribution (the human control in this case). We note our results are consistently closer to zero than normalised cut.

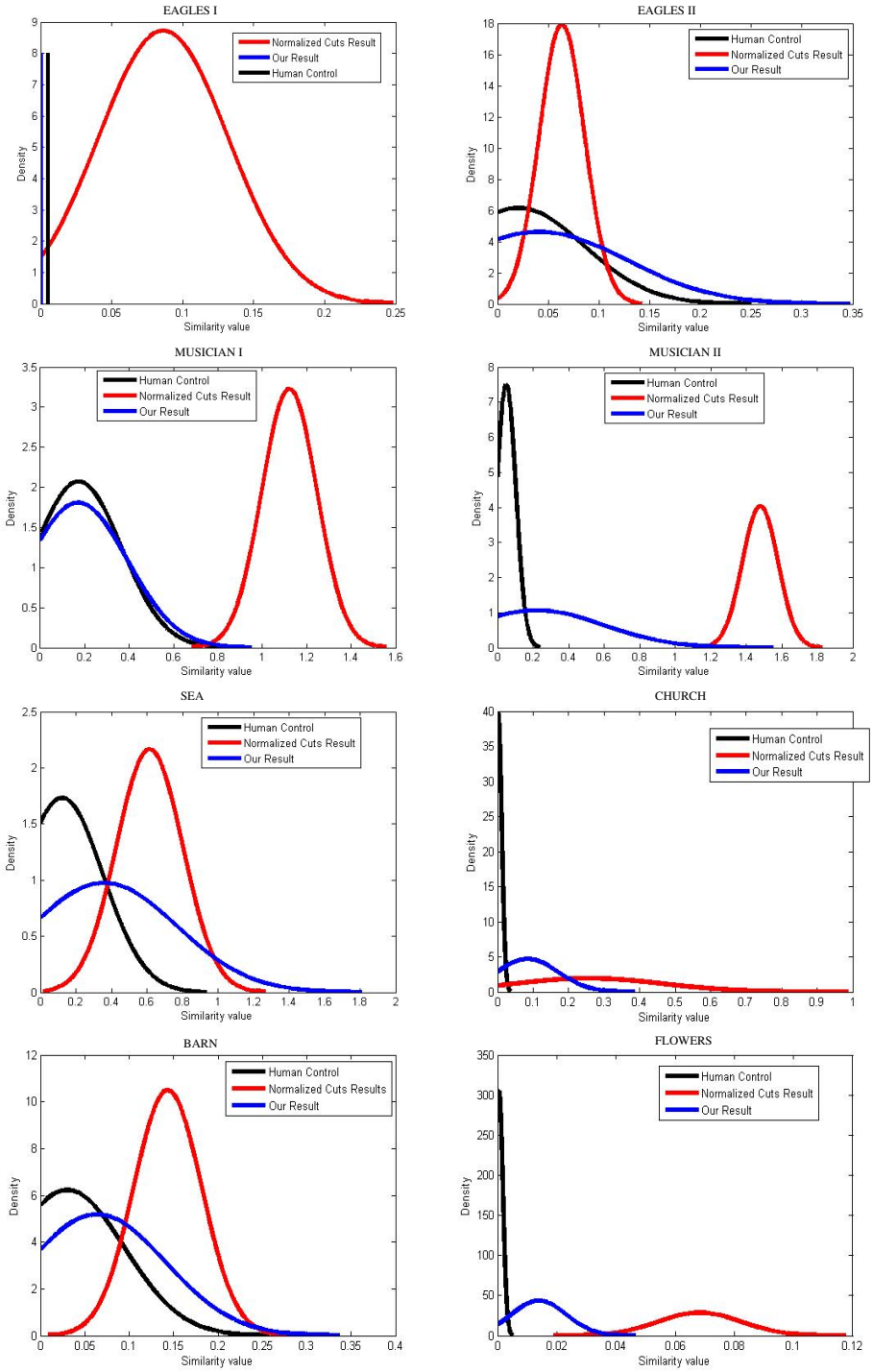


Figure 3-13: Gaussian distributions fitted to Laplacian graph similarities.

3.8.3 Experimental Conclusions

The result confirm that humans agree with one another more than they agree with automated groupers, which justifies our decision to use dissimilarity as a

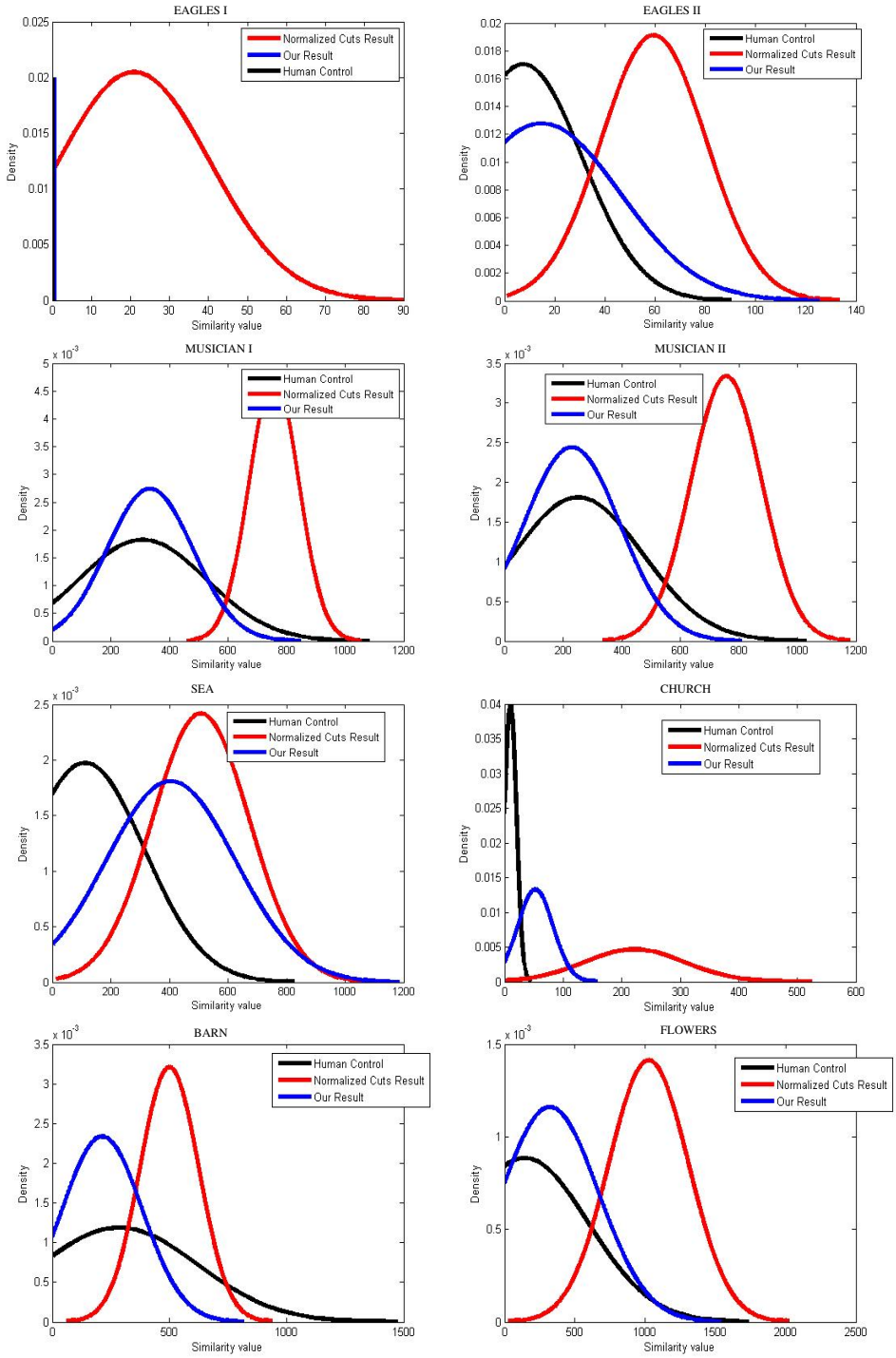


Figure 3-14: Gaussian distributions fitted to edit distance graph similarities.

base comparator. This can be seen in Tables 3.2 and 3.3, where human control results have the lowest similarity value and quite tight normal distributions.

In most cases our results are close to the human control, whereas results by

Table 3.4: Student T-test values for Laplacian graph similarities

	Human Control vs. Normalised Cut	Human Control vs. Our Method	Our Method vs. Normalised Cut
"Eagles" I	-5.97	0	5.97
"Eagle" II	-1.98	-0.60	0.80
"Musician" I	-13.12	0.019	11.89
"Musician" II	-40.27	-1.44	10.23
"Sea"	-5.28	-1.60	1.80
"Church"	-3.10	-2.10	4.70
"Barn"	-3.90	-0.60	2.70
"Flowers"	-7.90	-1.90	5.70

Table 3.5: Student T-test values for edit distance similarities

	Human Control vs. Normalised Cut	Human Control vs. Our Method	Our Method vs. Normalised Cut
"Eagles" I	-3.39	0	5.97
"Eagle" II	-5.24	-0.60	3.75
"Musician" I	-6.05	0.30	7.97
"Musician" II	-6.38	0.25	8.23
"Sea"	-4.79	-3.10	3.4
"Church"	-5.40	-1.00	3.90
"Barn"	-2.30	-0.90	1.50
"Flowers"	-2.80	-1.80	2.30

normalised cut are always worse, the T-test results proved numeric confirmation.

Overall, we interpret our results as meaning there is a wide range of groupings that humans find agreeable, but that groupings outside this range are heavily edited to fit within it. Depending on the image, humans disagree with our Prägnanz grouper to about the same extent as they disagree with one-another, but no matter how much they disagree with us they tend to disagree with normalised cut groupings more.

Some circumspection is required — we have presented detailed experiments on six images, but twenty-four different groupings in total. More experiments of this kind are needed, but are very expensive to run and preliminary experiments (on a different set of images) gave the same results. However, most people consis-

tently agree that our groupings are preferred over normalised cut groupings; the experiments confirm this.

3.9 Discussion and Conclusions

In this chapter, we offered a detailed explanation of the first of the two hierarchical image descriptions proposed in this thesis. This particular image description is based on perceptual grouping, specially, on finding stable and simple groupings of image primitives, i.e., Prägnanz. We have explicitly defined Prägnanz, for the first time, requiring groupings to be both stable and simple. Furthermore, our definition can be used as a controlling mechanism for any potential grouping algorithm, while not restricting to our own. A hierarchical image description is formed by organising the set of Prägnanz groupings returned from the grouper, where each layer of the hierarchy correspond to a particular "grouping scale". Finally, we showed that it is possible to pull out unique groupings over multiple "grouping scales".

In order to evaluate the performance of the proposed grouping technique, an experiment was designed and conducted. We approach this by comparing our grouping results to those of humans and normalised cut. It was shown that our method delivers groupings similar to those of humans and significantly outperformed normalised cut. In the next chapter, this experimental setup is also used to evaluate the performance of the second image description.

3.9.1 The Need for a Second Image Description

This first hierarchical image description has the advantages of (i) being able to produce multiple groupings for a single input image; (ii) such grouping can be organised into a lattice structure, so a hierarchical image description is formed; (iii) groups that often correspond to actual objects can be found using a salience measure.

Although proven via a quantitative experiment to perform well on a set of images, it can be observed that salient groups produced are often of middle scale. In other words, this particular grouper tend to find groups corresponding to whole objects

in a given image, rather than offering decompositions of such. This limitation is contradictory to our main goal of being able to extract object structures. It is mainly caused by the fact that only two Gestalt principles were used in the grouping process. Incorporating further Gestalt laws into the grouper will almost certainly improve its performance on extracting object parts, hence its structure. This is because those extra laws will introduce stronger ties to parts of the object, so that corresponding groups will appear more stable in the hierarchical image description.

In Section 3.4.1, we demonstrated that combining two Gestalt laws, proximity and common region, via logical propositions yield promising results. However, in the same section, we also demonstrated that combining Gestalt laws in an appropriate fashion is a difficult task. We have experimented with other means of logical propositions without any success. The main reason behind this challenge largely resides with that fact that there is still a lack of psychological evidence on how one Gestalt principle interacts with another. The study of relative importance among Gestalt laws is still regarded as an active research area by Gestalt psychologists. This difficulty of combining Gestalt laws can be treated as the bottleneck of the whole system. The performance of it, especially on finding object parts, will remain limited until the mutual interactions among Gestalt laws are better understood. Despite the fact that a more appropriate use of Gestalt laws is desired, one can still use our definition of Prägnanz to choose the best amongst many alternatives — and the sense of stability over grouping scale is likely to remain also.

It is this bottleneck on understanding the mutual relationship among Gestalt laws that leads us to propose an alternative solution. The next chapter offers a detailed description of the second hierarchical image description. It steers away from perceptual grouping towards agglomerative clustering. In particular, it aims to build up on the limitations of the first.

Another limitation of this first image description is that it does not operate equally well on grey-scale or binary images, which is a desired property, for example when we compare photos with line drawings. Also, it is unclear as to how this image description could be used for classification and painting.

Chapter 4

Hierarchical Agglomerative Clustering

In this chapter, we offer a detailed description of the second of the two hierarchical image descriptions this thesis proposes. This particular image description is not a built-up version of the previous one; but a completely new approach that aims to address the limitations of the previous description. Steering away from grouping image primitives using Gestalt laws, we build this second hierarchical image description by following an agglomerative clustering approach.

We demonstrate that by minimising a pair-wise cost function while merging image primitives, we are able to build a tree-like hierarchical image description. The bottom of the image hierarchy consists of basic image primitives, whereas the top node corresponds to the whole image. Each intermediate node in this hierarchy corresponds to a part of the image. This hierarchy is then parsed to yield semantic objects and their parts, either manually by specifying a single parameter or automatically using a novel graph theoretic measure.

We evaluate the performance of this second image description using the same experimental setup conducted on the first image description.

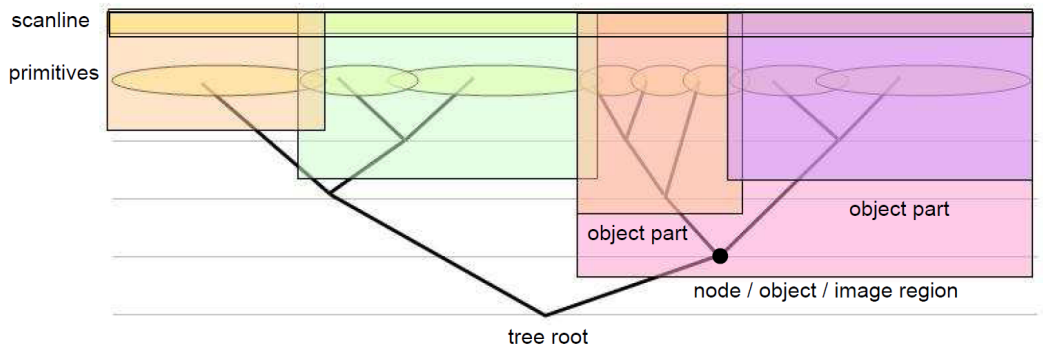


Figure 4-1: A diagram of our image description, in one-dimension. A scanline is broken into patches of different scale. These are merged into a binary tree. Each node of the tree specifies a particular section of the image. The tree is broken into nodes corresponding to objects, and objects can be broken up into their constituent parts.

4.1 Overview

This particular hierarchical image description is based on agglomerative clustering of image primitives. We start by choosing a set of image primitives, which form the bottom level of the image hierarchy. In this image description, we take image regions to be image primitives, so that image partitions are automatically formed once groups are decided. Those image region primitives are merged in a hierarchical fashion to form larger patches, until there is only one patch left, one that corresponds to the whole image. At each agglomeration step, a pair of regions are merged. The decision on which pair of regions to merge is decided by a cost function measuring the error of such merge. An image hierarchy containing all region primitives and the merged regions is formed in this way. Because of the binary nature of the merging process, the resulting image hierarchy can be treated as a binary tree, leaves of which correspond to the set of image primitives and intermediate nodes correspond to the newly formed regions. Figure 4-1 offers a 1-D visualisation of the type of image hierarchies generated in this fashion.

Having such an image hierarchy, the idea is then that objects and their parts will be nodes, each of which is a collection of image primitives. However, there are two major difficulties with this assumption. First of all, the question of whether objects and their parts exist in the hierarchy has to lot to do with the quality of the grouper. Secondly, we also need a way to identify such objects/parts from a relatively large number of nodes in the hierarchy. We offer solutions to both in our merging technique.

On the subject of choosing image primitives, initially, relatively simple region primitives such as regions defined by the watersheds of an image were used. Later, we will show by experiment that an alternative type of image primitives called, Difference of Gaussian (DoG) regions, offer a better foundation to the image description. DoG regions are circular image patches centred on a pixel, whose radius is determined by the first extremum of a DoG filter. Despite the fact that more appropriate image primitives can improve the overall quality of image descriptions, because of the way each primitive is described (explained later in this section), our hierarchical agglomerative clustering technique works regardless of primitive types.

Each image primitive is modelled as the distribution of feature vectors describing the region. A merging tree is formed by merging the most similar pair of neighbouring primitives at each agglomeration step. However, dimensions of the feature vectors are often correlated, which largely affect the performance of the similarity measure. We tackle this problem by decorrelation using Independent Component Analysis (ICA). A feature decorrelation matrix K, which is learnt under supervision, is introduced to bring each dimension of the feature vectors into their statistically independent directions.

Once a hierarchy is formed, the user is able to specify the number of desired parts by setting a single parameter in the system. This works because at each agglomeration step, two regions are merged, hence producing a different segmentation of $N-1$ segments, N being the number of primitives. For example, at agglomeration step 3, the image will be segmented into exactly $N-3$ parts. Hence, by specifying a parameter, the user can select the corresponding segmentation at a corresponding agglomeration step. Automatic means of parsing the image hierarchy is also possible, based a novel technique based on graph theory. This automatic parser acts to find the agglomeration step that is the most stable; hence outputs a segmentation of the image.

In the following sections, we will explain the merging algorithm in detail. Specifically, in Section 4.2, we test two possible ways of building image primitives for merging and experimentally show one is better than the other. The actual merging process is then explained in Section 4.3, followed by an quantitative experiment in Section 4.4. Finally, the chapter is concluded in Section 4.5.

4.2 Choosing Image Primitives: Watershed Regions vs. DoG Regions

As we previously mentioned, this second image description is formed hierarchically by merging image primitives using a bottom-up approach. Therefore, the very first task is choosing an appropriate set of primitives.

Normally, image primitives should conform to two general conditions: they should collectively form a covering set of the image, otherwise the final image description would be incomplete; they should be homogeneous regarding to some common image properties, so that merging makes sense. However, an extra condition is crucial in our case. That is, our image primitives should perform equally well on images of different depictions, i.e, photographs, paintings and drawings. Importantly, this invariance captured by image primitives will be carried into the final image descriptions.

However, many of the commonly used features, such as MSER [93], do not operate effectively over images exhibiting a wide range of depictive styles. An explanation is that the detectors were designed for and tested on continuous grey level or full colour photographs [101]. But drawings can be represented by binary images, and simple painting styles may use large areas of flat colour.

It is clear that feature detection across all depictions is not as straight forward as re-using existing literature. We considered two alternatives: watershed regions and DoG regions.

4.2.1 Watershed Regions

A watershed region is defined as an image patch covered by one and only one watershed generated using an appropriate watershed transform. They are attractive because they comply with all three desired properties set above: they cover the entire image; they are homogeneous in that each watershed has one and only one local minima to it; more importantly, they act equally well on images of all kinds. Moreover, we observed that they tend to isolate features of interest thought to be important for image description: dots, T-junctions, line ends, corners, contrast edges and so on. Watershed regions also tend to be of an appropriate scale.

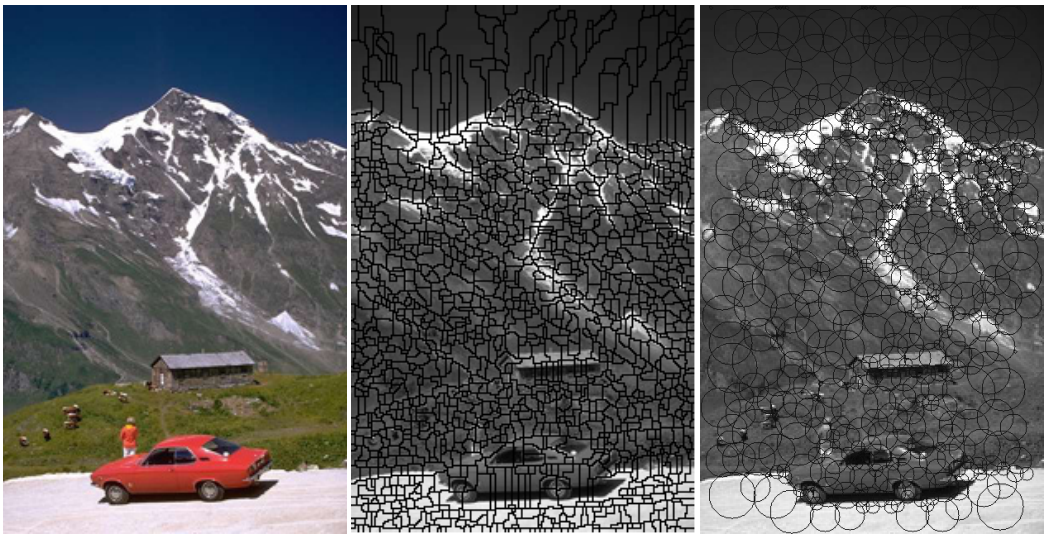


Figure 4-2: A comparison between watershed regions and DoG regions on the same images. Left: original image; middle: result of performing watershed transformation on the corresponding intensity image; right: DoG regions overlaid on the original image.

Please note that unlike work aimed to segment images, see [59, 48] for example, our watersheds are computed directly from the intensity image, not from its derivative magnitude. There are many watershed transformation algorithms in the literature, we chose to use the "Fast Watershed Transform" by Vincent and Soille [167]. We chose this particular algorithm largely because of its computational efficiency and the fact that it is readily available in Matlab. Vincent and Soille's watershed algorithm works by mimicking an immersion process, where water flooding are modelled by a queue of pixels. The speed of the whole process is extensively improved by sorting pixels in terms of their intensity values, prior to the actual flooding step. In the middle of Figure 4-2, we illustrate the result of applying "Fast Watershed Transform" to the colour image on the left.

4.2.2 DoG Regions

DoG filter is particularly attractive for our needs. It lie behind SIFT descriptors [89], and therefore by transitivity lie behind much of the visual object learning literature. One motivating force for this is that DoG filters have been experimentally determined as providing excellent estimates of local scale [101]. They operate as follows. Let $g(x|\sigma)$ be an origin centred Gaussian filter of width σ, and let $d(x|\sigma_1, \sigma_2) = g(x|\sigma_1) - g(x|\sigma_2)$ be a DoG. Typically these scale values are

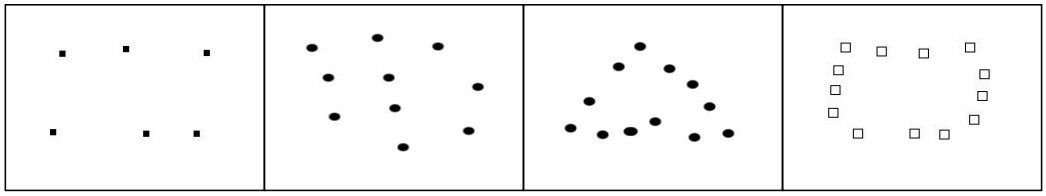


Figure 4-3: Four hand-drawn images used in the experiment

related in some way such as $\sigma_1 = \sigma_2^2$, so that we may write $d(x|\sigma)$ for a DoG. Let $h(x|\sigma) = f(x) * d(x|\sigma)$ be a filtered image. The function $h(x|\sigma)$ as σ varies is the response signal at a pixel. The extrema of this signal occur at scales values ρ; they indicate the existence of a local feature centred at the point x of scale ρ. The value $|h(x|\rho)|$ is the strength of the response.

Because DoG filters give a set of scale values at every pixel, they can also be regarded as producing a set of scaled circular patches at every pixel. These patches enclose the local feature centred on the pixel. We used the smallest scale, ρ, at every pixel, which gave us a scale map. We filter this scale map to produce a final set of image primitives. First, any regions that extend over the limits of the picture are discarded. Second we use a greedy algorithm to choose a subset of salient patches. We add the most salient remaining region to the kept list (initially empty) and discard all regions with a centre lying inside this region. We follow Collomosse and Hall [20] to determine salience. Then repeat the above steps, until all regions are kept or discarded. The result is a set of circular regions, each scaled to the locally smallest visual feature. These overlap to form a dense covering over the image, and every region is wholly contained within the image.

An example of DoG regions with comparison to watershed regions is given in Figure 4-2, where the right image demonstrates the set of DoG regions found on the corresponding colour images on the left.

Like watershed regions, DoG regions also satisfy all three requirements of merging primitives set towards the beginning of Section 4.2.

4.2.3 Choosing Between Watershed and DoG Regions

Having two types of primitive, DoG regions and watershed regions, we must choose between them. The basis of the choice was the consistency of our region descriptor, for it is important that regions containing the same feature are consistently described. Given a region of any shape, we describe it by the statistical distribution of measures taken at every pixels it contains. We could use any measure, but confined ourselves to colour and image gradient magnitude. Furthermore, we assumed the measures to be Gaussian distributed in feature space. This feature space should not be confused with the structural features which are our ultimate objectives and which occupy a different feature space.

We conducted an experiment to measure the variance of region descriptions. We prepared four simple images: black dots on a white background (shown in Figure 4-3). Thus each feature is an identical dot and so should be identically described. In this experiment, our descriptor reduced to grey level mean and variance. The test images were processed to obtain DoG regions and watershed regions. We then computed the variance of the descriptors, which ideally should be zero given each visual feature was identical. In practice, the content of surrounding image has an influence on the size (and for watershed, the shape) of the regions, so the variance of description is not zero. It is worth noting that more images could be used in this experiment, such as these consisting of more identical features in different spatial layouts. However, we feel four is sufficient in our case as we can measuring the variances in the feature descriptions, so that more features would not necessarily affect the outcome.

The detector with the most consistent descriptions turned out to be the DoG algorithm with the greedy filter approach; it offered an order of magnitude improvement over watershed regions.

Having experimentally justified the use of DoG regions, in the rest of this chapter, we will use DoG regions as the default choice of merging primitives. As an early evidence, Figure 4-4 offers two side-by-side examples of merging results from the same grouper using watershed regions and DoG regions, respectively. It can be seen that between the two, DoG region merging offers better segmention. In the following section, we will continue to explain the merging method that was used to produce such results.

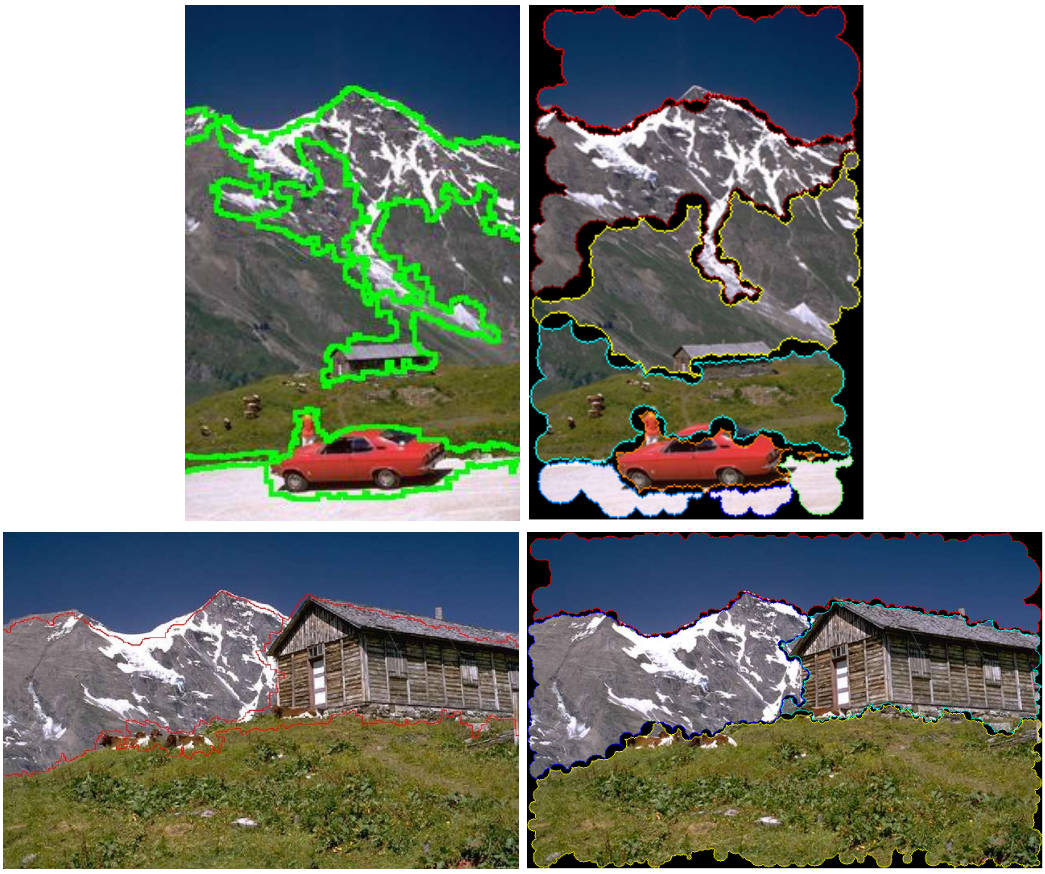


Figure 4-4: From left to right: merging results using watersheds and DoG regions, respectively.

4.3 Merging Method

Having defined our image primitives, we can continue to merge them in a hierarchical fashion in order to form an image description. In this section, we explain the agglomerative clustering algorithm that we use to merge image primitives. As previously mentioned, our grouper is independent of the type of primitives. The reason for this will become clear later in the section.

In general, we generate this hierarchical image description by agglomerative clustering of neighbouring primitives. We define a pair of primitives to be neighbours if they overlap. A pair of neighbouring regions are merged to create a new region, which in turn may be merged to its neighbours and so on. Eventually only one region remains. Each merging event creates a node in the binary tree; the node contains the newly formed region and the branches of which point to the components. A one-dimensional visualisation is shown in Figure 4-1.

Our merging method is based on the work of Haris et al. [59] and Malpica et al. [90]. Haris et al. defined the cost e_{ij} of merging a pair of regions i and j as

$$e_{ij} = \frac{N_i N_j}{N_i + N_j}(\mu_i - \mu_j)^2 I(i,j) \tag{4.1}$$

where μ_i and μ_j are mean intensity values in each region of sizes N_i and N_j, respectively; and

$$I(i,j) = \begin{cases} 1, & \textit{if regions } i \textit{ and } j \textit{ are adjacent} \\ +\infty, & \textit{otherwise} \end{cases}$$

At each merging step, adjacent regions with the smallest e_{ij} are merged, creating a new region from the union of the regions and their boundary (a greedy algorithm). This process continues until the cost exceeds a pre-computed threshold which is based on the overall noise distribution of the image. However, we note that this threshold is image dependent and that noise is difficult to estimate and often unreliable when dealing with more complex images than those synthetic and medical ones used in their paper.

Malpica et al. [90] extended Haris et al.'s algorithm by allowing a multi-variate description for each watershed region, other than using region mean intensity alone. Their extension uses the mean of feature vectors in a region, so μ is now a vector too. They redefined the cost as being proportional to the sum the root of absolute differences of individual elements between means: $\sum_k |\mu_{ik} - \mu_{jk}|^{1/2}$, the scale factor $N_i N_j/(N_i + N_j)$ is the same.

We differ from Malpica et al. [90] in two important ways:

- We model image primitives as distributions of feature vectors, rather than its mean
- We demonstrate the value of decorrelating the feature vectors

Each pixel $\mathbf{x}$ in region, $\mathcal{R}_i$, supports a vector of measures $\mathbf{v}(\mathbf{x})$, perhaps via filtering. We simplify the distribution of features vectors in a region with a

Gaussian: N_i – the number of pixels, μ_i – the mean, and $\mathbf{C}_i$ – the covariance:

$$N_i = |\{\mathbf{x} \in \mathcal{R}_i\}| \tag{4.2}$$

$$\mu_i = \frac{1}{N_i} \sum_{\mathbf{x} \in \mathcal{R}_i} \mathbf{v}(\mathbf{x}) \tag{4.3}$$

$$\mathbf{C}_i = \frac{1}{N_i} \sum_{\mathbf{x} \in \mathcal{R}_i} (\mathbf{v}(\mathbf{x}) - \mu_i)(\mathbf{v}(\mathbf{x}) - \mu_i)^T \tag{4.4}$$

Together these terms make up an eigenmodel to represent the region. The encoding error in this approximation is

$$\mathbf{H}(\mathcal{R}_i) = \sum_{\mathbf{x} \in \mathcal{R}_i} (\mathbf{v}(\mathbf{x}) - \mu_i)^T (\mathbf{v}(\mathbf{x}) - \mu_i)$$

The encoding error for the whole picture is the sum of errors in each primitive region.

$$\mathbf{W}(\mathbf{P}) = \sum_{\mathcal{R}_i \in \mathbf{P}} \mathbf{H}(\mathcal{R}_i)$$

where $\mathbf{P} = \mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_k$ is a k partition for the image.

The optimal partition minimises the encoding error. Using a greedy approach, we merge the pair of regions that gives minimal cost. The cost of merging a pair of regions is

$$e_{ij} = \frac{N_i N_j}{N_i + N_j} (\mu_i - \mu_j)^T (\mu_i - \mu_j) \tag{4.5}$$

when two regions are merged we can efficiently compute a new Gaussian to approximate the distribution of the features in their union:

$$\mathbf{C}_{\text{new}} = \frac{N_i}{N_i + N_j} \mathbf{C}_i + \frac{N_j}{N_i + N_j} \mathbf{C}_j + \frac{N_i N_j}{(N_i + N_j)^2} (\mu_i - \mu_j)(\mu_i - \mu_j)^T \tag{4.6}$$

The final term allows for the difference between means. It can be thought of as measuring the volume of the symmetric difference of two classes. Its trace is proportional to the error as we define it, and is also related to the errors defined by Haris et al. [59] and Malpica et al. [90].

It appears that overlapping regions make a difficulty, because pixels are counted more than once. But since it can be shown that using the above to merge two regions with an identical distribution yields the same distribution, it is possible

instead to simply adjust the number of points in the merged region to record its true area.

One might argue that, as the regions merge the feasibility of the Gaussian assumption will decline. This is acceptable because we are willing to accept broader approximations of larger regions, which is in line with scale-space descriptions.

4.3.1 Decorrelating Feature Vectors

By now, the description of our merging method is complete. The method described works reasonably well, but its performance is expected to improve by taking out any correlations between elements in the feature vector. This is because the vector $\mu_i - \mu_j$, which is central to the merging error, assumes statistically independent directions.

It may seem that the eigenmodel should already decorrelate feature vectors, as we place no restriction on the covariance matrices. However, the eigenvectors arising from eigen-analysis do not necessarily give statistically independent directions — independent component analysis (ICA) [67] is needed for that.

ICA is a generative statistical model that assumes a linear dependency between independent terms. The independent components are latent variables, which cannot be directly observed. ICA sets off to find such statistically independent terms from a collection of observed data. Assume we have observed n linear mixtures $\mathbf{x} = x_1, ..., x_n$ of n independent components $\mathbf{s} = s_1, ..., s_n$

$$x_j = a_{j1}s_1 + a_{j2}s_2 + ... + a_{jn}s_n, \text{ for all j}$$

which can be re-written as

$$\mathbf{x} = \mathbf{A}\mathbf{s}$$

using vector notation. Here, $\mathbf{s}$ is a vector of hidden variables, which cannot be directly observed; the mixing matrix $\mathbf{A}$ is also unknown. All we observe is the random vector $\mathbf{x}$, and we must estimate both $\mathbf{A}$ and $\mathbf{s}$ using it. ICA is able to do exactly this for us. After estimating the matrix $\mathbf{A}$, we can obtain the independent component simply by:

$$\mathbf{s} = \mathbf{W}\mathbf{x}$$

where $\mathbf{W}$ is the inverse of $\mathbf{A}$.

In our case, we too want a decorrelating linear transform, a square matrix $\mathbf{K}$, so that

$$\mathbf{x} \mapsto \mathbf{K}\mathbf{x} \tag{4.7}$$

and the new de-correlated feature vector $\mathbf{K}\mathbf{x}$ is to be used in the grouper instead. Now the error in Equation 4.5 is proportional to

$$(\mu_i - \mu_j)^T \mathbf{K}^T \mathbf{K} (\mu_i - \mu_j)$$

Since $\mathbf{x}^T \mathbf{A} \mathbf{x}$ can be written as an inner product $\mathbf{A}_{ij}(\mathbf{x}\mathbf{x}^T)_{ij}$ (using tensor notation) we see that the error is just a linear combination of all the quadratic terms in the difference between the means, and not just the diagonal.

We approach estimating the matrix $\mathbf{K}$ using a supervised learning paradigm. The main idea is to collect a sufficiently large amount of observed data that are correlated. Using such data we can then obtain the decorrelation matrix $\mathbf{K}$ using ICA. This $\mathbf{K}$ matrix can later be used to decorrelate unseen data. A total number of 10 testing images from the Berkeley segmentation database [92] were used for the training purpose. We first compute primitives for each image. Users were then asked to pick pairs of adjacent regions they wish to be merged using a simple GUI. During training, a user may merge in a hierarchical fashion. We recorded a total of 1500 such pairings and thereby obtain a training collection of $\mathbf{u}_{ij} = (\mu_i - \mu_j)$ vectors as our observed data. Given the set of input training vectors $\mathbf{u}_{ij} \in \Re^n$, we determine a $n \times n$ matrix $\mathbf{K}$ using ICA, such that the components of the mapped vector $\mathbf{K}\mathbf{x}$ are mutually independent.

The feature de-correlation matrix $\mathbf{K}$ is computed using a fixed-point based variant of ICA due to Hyvärinen [67] known as "Fast ICA"; which is computationally very efficient yet statistically robust. The actual value of $\mathbf{K}$ is given below.

$$\mathbf{K} = \begin{bmatrix} -16.2199 & 20.4293 & -9.9570 & 1.7411 \\ 2.4867 & 9.5783 & -19.0278 & 0.4898 \\ 1.8215 & 13.3457 & -7.5742 & -3.2843 \\ 0.0602 & -0.7559 & -0.1765 & 9.6223 \end{bmatrix}$$

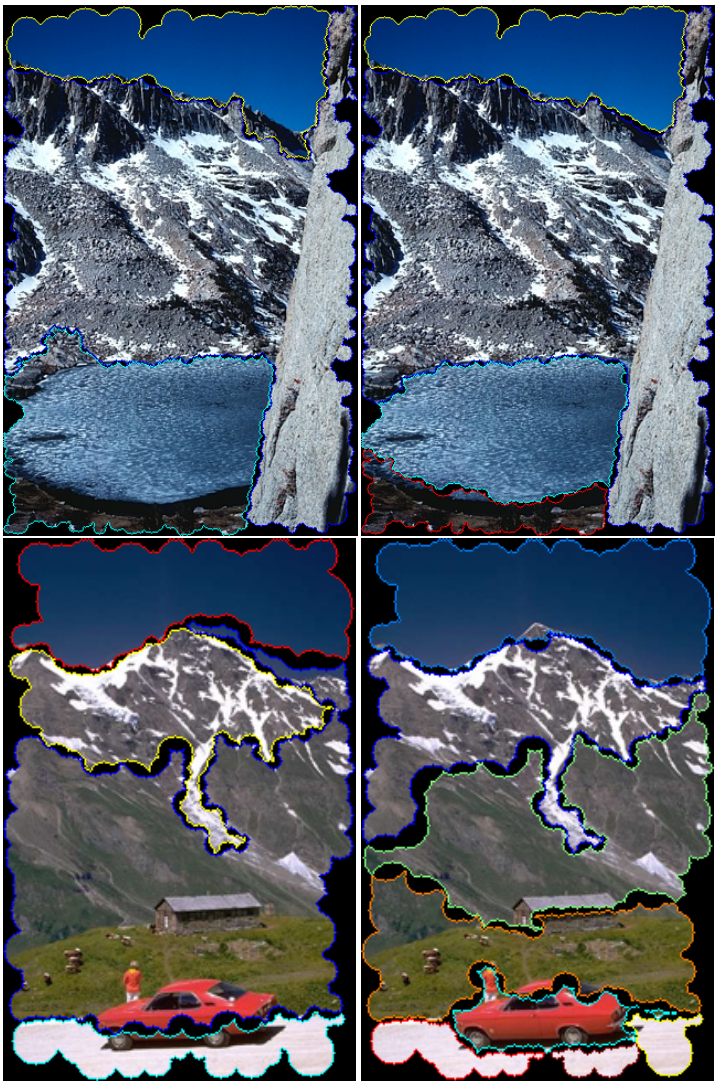


Figure 4-5: From left to right: DoG region merging results without and with decorrelating feature vectors, respectively.

To recall, the role of K is to map any characterising vector into statistically independent axes. The rationale for this is that decorrelating a data set along statistically independent axes will lead to improved error measures. Region merging results in Figure 4-5 demonstrate the value of the decorrelation matrix, K. For example, on the top row, the mountain on the right and the lake are better segmented; on the bottom row, the car and the hut are more clearly delimited using K.

4.3.2 Manual and Automatic Selection of Nodes

Given an image, we can now build its image hierarchy in the above described fashion. Within the binary tree like hierarchy, each leaf node is a single primitive, i.e., a DoG region; whereas, each intermediate node corresponds to a particular part of the image and the root node represents the image as a whole. Our task now is determining ways of locating meaningful parts within such image hierarchies, with the hope that the filtered parts will correspond to semantic entities in the image.

We offer two ways of locating meaningful nodes from image hierarchies: manually by specifying a single parameter indicating the number of nodes required and automatically, by appeal to a novel graph theoretic measure. The latter approach can also be treated as means of determining the user-specifiable parameter automatically.

A typical merging process works as follows. From an input image, we first determine a finite set of S DoG regions as merging primitives, this can also be treated as a segmentation with $|S|$ regions. Such primitives are then merged using the grouper described in Section 4.3. At each agglomeration step, two regions are merged, resulting in a new segmentation of $|S|-1$ regions. This merging process stops when there is only 1 region left, that is, the original image. In general, at agglomeration step n, there will be $|S|-n$ regions in the resulting segmention. Moreover, because of the binary nature of the grouper, an image will take exactly $|S|-1$ agglomeration steps from start to finish. For example, at the last merging step $n=(|S|-1)$, we will be left with $|S|-n=|S|-(|S|-1)=1$ region(s), which is the original image. Figure 4-6 illustrates a few segmentations along a typical merging process.

Manually Selecting Nodes from the Hierarchy

Having observed the above characteristic of the grouper, it seems natural to use n, the agglomeration step, as an user-specifiable parameter to pick parts from the image hierarchy. However, n is related to the number of primitives which is image dependent and is often large, hence not user-friendly. A more suitable and intuitive parameter would be the number of segments that an user wants, i.e., $|S|-n$. We therefore set $\sigma=|S|-n$ as the only parameter that an user can

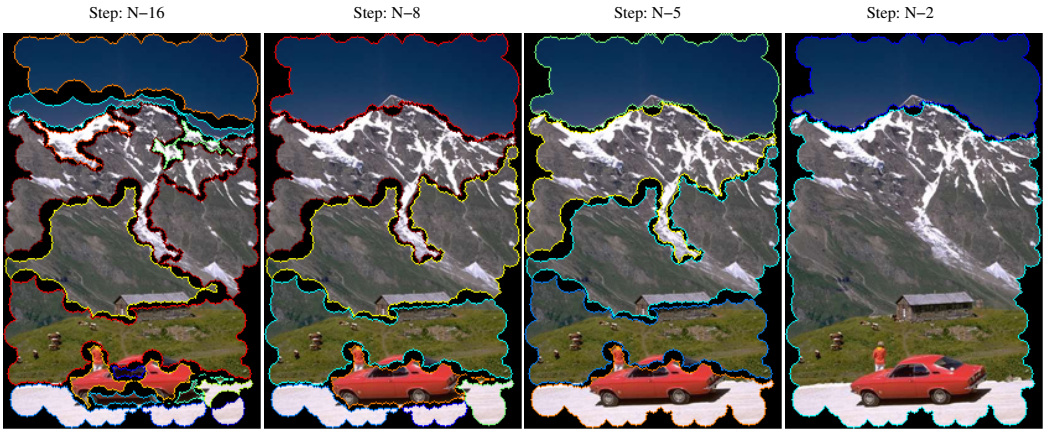


Figure 4-6: Segmentation results along the merging process. Left to right: segmentations at agglomeration step, $N-16$, $N-8$, $N-5$ and $N-2$, respectively.

choose, where σ corresponds to the extact number of parts that is output. Given a value of σ, we can always map back to the corresponding agglomeration step by $n = |S| - \sigma$, hence obtaining the required segmentation at n. It is not hard to see that σ is just a more intuitive equivalent of n. Our parameter, σ, is much more intuitive comparing with that of Haris et al. [59]. They used hard threshold based on a global image noise estimation, which is image dependent and doesn't work so well on natural images.

Two sets of examples are provided in Figure 4-7 for qualitative inspection, where the user chose to segment the images in different ways by setting σ accordingly. It can be seen that the proposed grouper works quite well on both the photograph of a face and that depicted in a line drawing.

Automatic Node Selection using Graph Energy

Similar to the manual method, an automatic algorithm should aim to identify a n, specifying a particular agglomeration step. We propose such an algorithm based on spectral graph theory [17]. In particular we modify the definition of the Laplacian graph energy [53]. Before we explain what this means we should remind ourselves briefly on how the image hierarchy was built, in terms of its graph representation.

The hierarchy is a binary tree. Any node in the tree has a subtree beneath it. The leaves of the subtree are primitives that cover some area of the image surface.

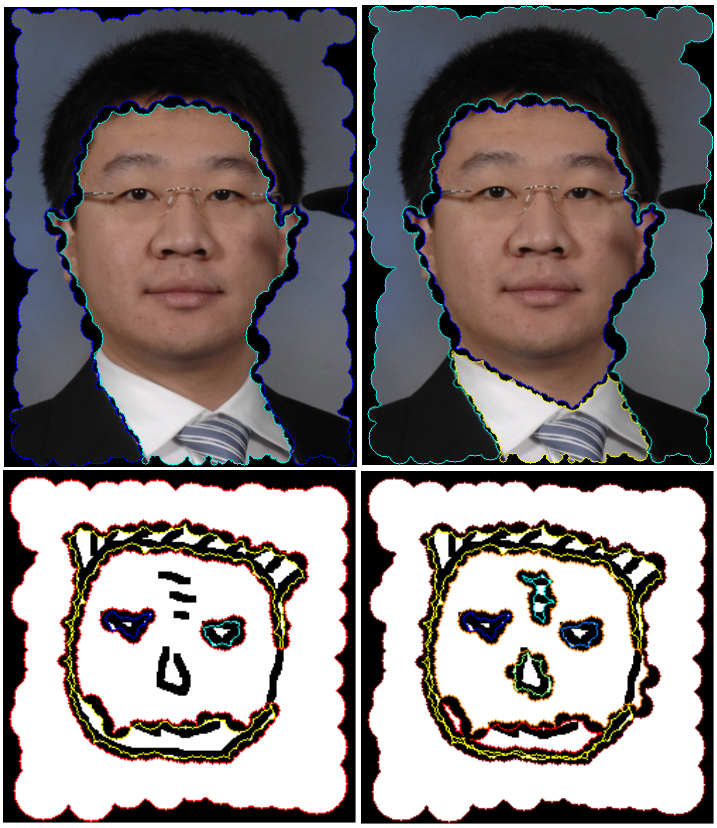


Figure 4-7: Top left to top right: merging result of a photograph when $\sigma = 2$ and $\sigma = 3$, respectively; bottom left to bottom right: merging result of a line drawing when $\sigma = 4$ and $\sigma = 8$, respectively.

These primitives are connected into a graph using neighbourhood relations. So, any node in the tree can simultaneously be considered as a subtree, an image region, and a graph. More precisely, any node corresponds to a subgraph of the graph from the root node, which contains every primitive. It is the subgraphs made by considering neighbour relations between primitives that are of interest in this section.

It is important to recognise that although the region covered by a set of primitives has properties such as average colour, and so on, we make no use of them. This is because we wish to assume as little about the image content as possible. Instead we base our analysis entirely on graph properties, spectral graph energy in particular.

Now, consider a subgraph which is fully connected. Such a subgraph is likely to correspond to a coherent object. In our case we happen to know that coherence over regions is built-in to the construction process because the regions are merged

so as to minimise error over the property distribution; but this information is not explicitly used in this section. In any case, coherent regions are regarded as salient. The intuition is that intra-region region nodes "pull together" more strongly than inter-region nodes. This intuition is one which is already exploited by graph cuts [142]. Unlike graph cuts, our algorithm automatically chooses the number of image segments.

Similarly, subgraphs that are cyclic (i.e. can be drawn down as a polygon) are salient — they correspond to polygons, for example. Cyclic relations are also salient to image understanding. Gestalt tells us such cycles produce "pop out" visual feature [73]. We wish to parse the tree into subtrees that correspond to subgraphs with properties of this kind, and can do so using graph spectral theory, as explained next.

Laplacian graph energy has the following standard definition. Let $G = (V, A)$ be an undirected graph of with nodes N and arcs encoded by the adjacency matrix of a graph. The degree of any node is the number of edges associated with it, and D is a diagonal matrix holding the degree of each node. The Laplacian matrix is then defined as $L = D - A$. Let $w(i, j)$ be the some weight associated with the arc (i, j). We set $w(i, j) = 1$ if there is an arc between nodes i and j, and 0 otherwise. It would be possible to use data gathered from the construction process, such as the error term from Equation 4.5 and set $w(i, j) = exp(\alpha e^2)$ for some constant α. Either way, the Laplacian graph energy [53] has the following standard definition

$$\mathcal{E}(G) = \sum_{i=1}^{|V|} \left| \lambda_i - \frac{m}{|V|} \right| \tag{4.8}$$

in which the λ_i are eigenvalues of the Laplacian matrix and m is the the sum of the arc weights over the whole graph.

In practice, inevitably, there are merging steps when the description contains disconnected components. We allow for this, as a result, we modify the Laplacian energy to

$$\xi(G) = \frac{n}{N} \sum_{i=1}^{N} \frac{\mathcal{E}(G_i)}{|V_i|} \tag{4.9}$$

as the graph energy of the tree at any stage in its construction. In this N is the number of distinct connected components and $|V_i|$ is the number of nodes in the

ith component. The term $\mathcal{E}(G)/|V|$ is the average connection energy per node. The factor n is the number of nodes in the whole graph — the number of image primitives. It is used only to ensure our re-definition of graph energy returns to the original Laplacian definition when there is a single connected component (at which stage $N = 1$ and $|V_1| = n$).

Figure 4-8 shows our graph energy plotted as a function of the number of steps the merging algorithm has taken. The general rise in energy is explained by the rise in the number of arcs. Local minima occur when disconnected components become more fully connected, or move toward a cyclic graph. As construction of the merging tree proceeds, the number of disconnected components falls. Therefore we parse the whole tree by selecting the last merging step to exhibit a local minimum in graph energy ξ. It is worth noting here that other local minimas also correspond to specific segmentations of the image, however, the usefulness of these are deemed as future work.

To summarise, during a merging process, each step yields a graph energy, i.e., the partition energy is a function of iteration number. We continue merging until just one group remains, but back-track so as to halt at the local minimum of ξ with the largest iteration count. As previously explained in Section 4.3.2, an agglomeration step corresponds to a particular segmentation of the image and can be manually selected using a single parameter σ. Graph energy also yields an agglomeration step that it finds its last local miminum at, hence can be treated as an automatic way to set σ.

Graph energy has already been used to produce some of the results presented in this chapter, such as Figure 4-4, 4-5 and the right column of Figure 4-7. More results are available in Chapter 6 of this thesis, where graph energy is used in a recursive fashion to extract object structures.

4.4 Quantitative Evaluation

In this section, we conduct an quantitative experiment to evaluate the performance of this second image description. The same experimental method used to evaluate the Gestalt-based image description (Section 3.8) is employed for this experiment. In particular, we compare the newly proposed merging technique

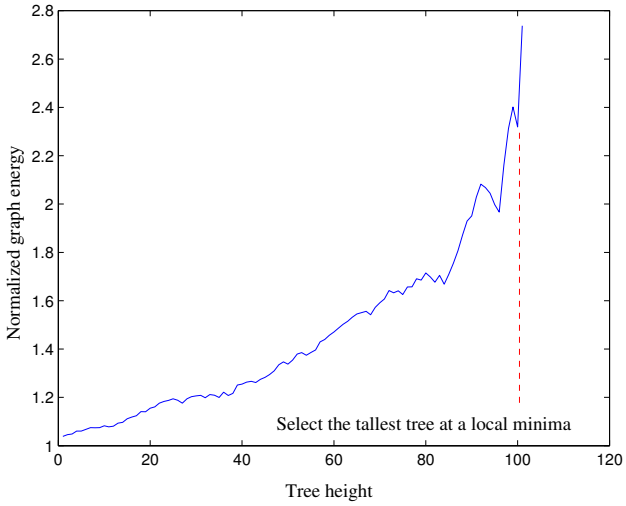


Figure 4-8: An example of normalised graph energy as a function of tree height. We identify an object using the local minimum with tallest tree.

with the Gestalt-based grouper proposed in the previous chapter and both image descriptions with that of humans. Both of these two groupers require a single input parameter, which is the number of groups required in the final output.

More specifically, we use exactly the same set of test images used in the previous experiment (shown in Figure 3-11). Both human and Gestalt-based grouping results are kept in this experiment (2nd and 3rd column of Figure 3-12). We simply replaced all Normalised Cut groupings with those produced using the newly proposed DoG region based merging technique. To be consistent, the number of regions output by the DoG-based method is set manually using the technique described in Section 4.3.2, in accordance with what was previously used to produce Gestalt grouping results. To convert image regions output by the DoG grouper to line groupings, we again overlay the same set of line segments on top of regions returned by the new DoG-based method. All 24 groupings used in the experiment are provided in-turn in Figure 4-9, where each column corresponds to a particular grouping technique and each row has a common number of groups. The right most column shows all 8 new groupings output by the new method. After some quick observation, it is not hard to see that the new DoG grouper tends to produce sensible groupings on all the testing images. It lives up to the good performance of Gestalt grouper on the simple "eagles" image, and tends to outperform on all other images, especially on "sea" and "flowers".

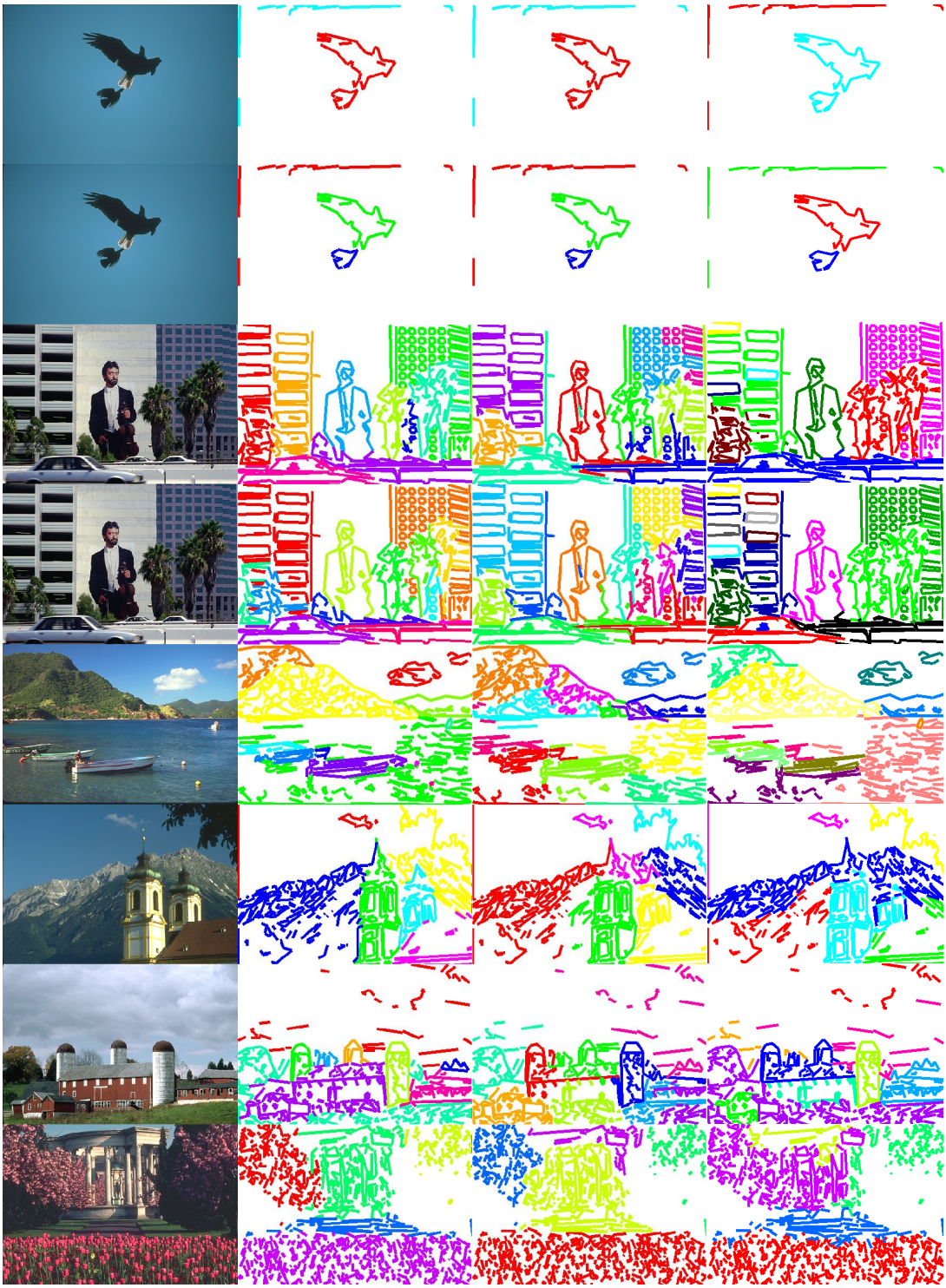


Figure 4-9: All 15 grouping used in the experiment. Columns left to right are: original image, human Control, Gestalt grouper and DoG grouper. Each row is unique way of forming groups in an image.

Table 4.1: Similarity measurement for different methods by using graph edit distance

	Human Control	Gestalt Grouper	DoG Grouper
Eagles I	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
Eagles II	0.00 (±0.00)	0.00 (±0.00)	0.00 (±0.00)
Musician I	241.56 (±187.30)	356.90 (±183.90)	329.32 (±202.02)
Musician II	365.10 (±123.32)	473.39 (±179.04)	415.08 (±190.13)
Sea	187.45 (±188.38)	442.50 (±194.78)	235.12 (±103.59)
Church	19.81 (±32.45)	56.39 (±64.06)	32.25 (±40.02)
Barn	205.06 (±158.97)	434.48 (±234.30)	352.19 (±197.78)
Flowers	93.52 (±246.73)	424.70 (±282.48)	188.41 (±335.44)

Table 4.2: Student T-test values for edit distance similarities

	Human Control vs. Gestalt Grouper	Human Control vs. DoG Grouper	DoG Grouper vs. Gestalt Grouper
"Eagles" I	0.00	0.00	0.00
"Eagle" II	0.00	0.00	0.00
"Musician" I	-2.15	-0.65	3.10
"Musician" II	-2.80	-2.23	1.32
"Sea"	-3.59	-1.41	7.45
"Church"	-1.63	-1.27	1.24
"Barn"	-2.47	-1.98	2.30
"Flowers"	-3.78	-0.97	8.15

In practice, we used 8 human subjects and the groupings are again presented to each subject in a random order known neither to the subject nor the experimenter but chosen by the software. Graph edit distances [122] are used as a measure of "semantic inefficacy": a greater distance is taken to mean a less efficacious grouping method. We didn't use Laplacian eigenvalues [17] as a distance measure, simply because both techniques were shown to provide identical results in the previous experiment (Section 3.8).

In Table 4.1 we show the mean similarity values for the three different grouping processes, computed using graph edit distance. Similar to that of Table 3.2 and 3.3, each table has three columns, one per grouping process. There are 8 rows to the table, each representing a fixed number of groups, so these tables correspond 1-1 with the images in Figure 4-9. In Figure 4-10, we show the same

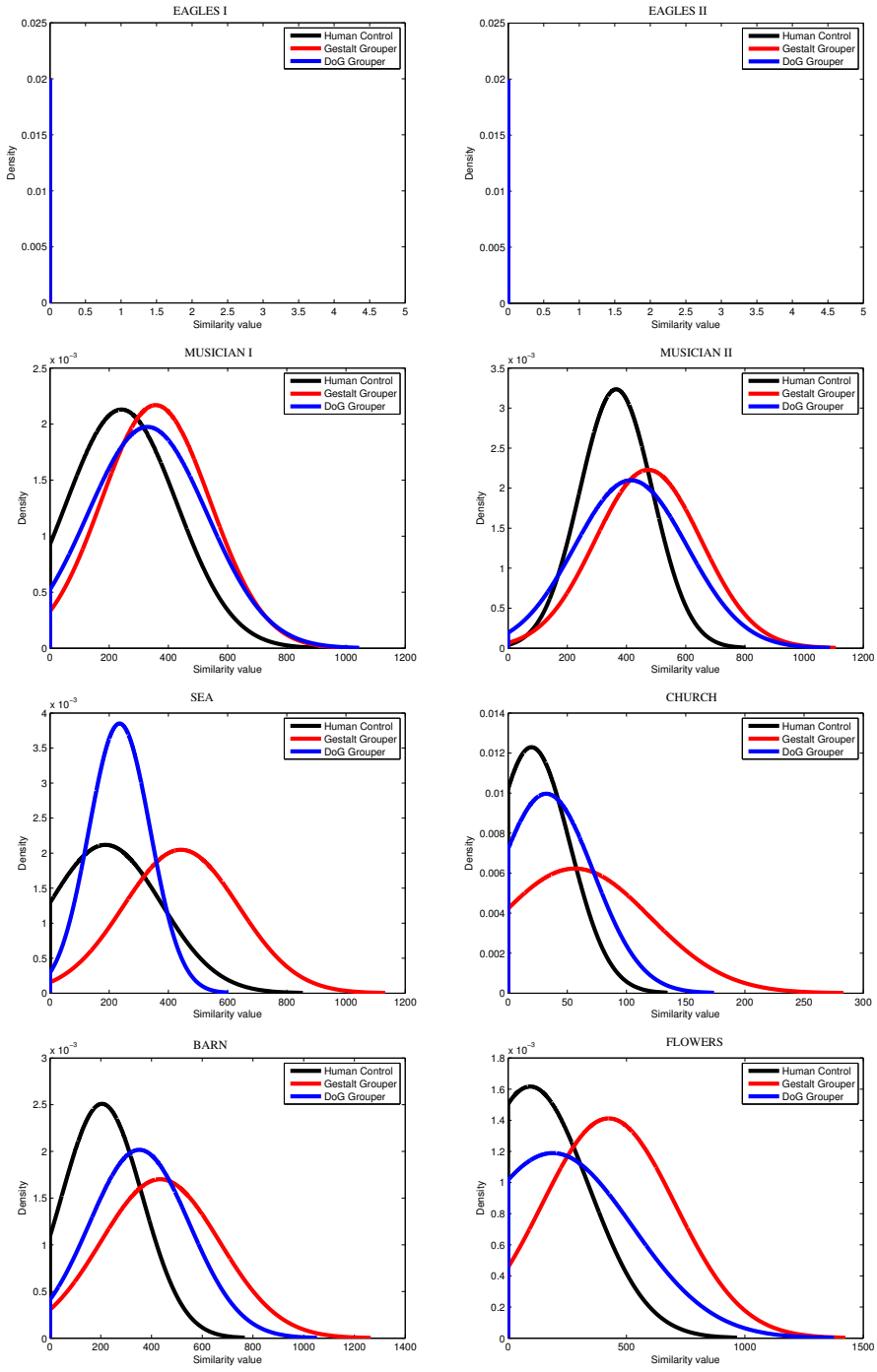


Figure 4-10: Gaussian distributions fitted to edit distance graph similarities.

data presented in graphical form.

As can be seen, on average users performed fewer edits on all DoG groupings than those output by the Gestalt grouper. In particular, it can be seen that there were significantly less average edits on "sea" and "flower" groupings using the DoG-

based method, than those using Gestalt grouper. Such observations can also be confirmed in Table 4.2, where the student T-test results on Gaussians derived from Graph Edit Distance similarities are provided. As mentioned before, a low absolute value of the T-test implies a closer distance between the distributions and the sign of the T-test locates the mean of the test distribution with respect to the reference distribution (the human control in this case). In addition, from these tables, we can confirm once more that humans tend to agree with others more than they agree with computers.

4.5 Conclusions

In this chapter, we proposed a second image description that aims to tackle the limitations of the first (Chapter 3). This particular image description is built using agglomerative clustering, where image primitives are merged by comparing the distribution of features vectors in adjacent pairs. The value of a trained decorrelating matrix is shown. The merging process is halted either by setting a user-defined parameter or by a picture independent criterion taken from graph spectral theory. A similar experiment to that used to evaluate the performance of the first image description was conducted and results of which confirmed the superiority of this second image description. In the rest of this thesis, we will use this new agglomerative clustering based image description regardless, unless indicated otherwise. In particular, we will demonstrate how objects and their topological structures can be extracted from this image description, which are later used to classify objects and produce abstract artworks.

Chapter 5

Interactive Editing

In this chapter, we describe how we make use of human interaction in the process of extracting object structures. More specifically, we implement an image hierarchy editor with intuitive graphical user interface (GUI) for users to interact with the automatically generated hierarchical image descriptions, especially of the type proposed in the previous chapter. The goal of the editor is that any user can use it to alter automatically generated hierarchies to the extent that the refined hierarchies correspond to the semantic topological decompositions of objects.

5.1 The Need for an Editor

It is desired to have object structures automatically extracted from images. However, to date, no computer vision algorithms can deliver this without prior models. Requiring prior models has two major drawbacks: (i) the user has to specify what models to use; (ii) there are a limited number of models to choose from and acquiring such models often needs lengthy supervised training. We, however, want to be able to extract object structures in an unsupervised manner and from any image the user supplies.

We proposed two hierarchical image descriptions for the purpose of extracting objects structures. Using the same experimental setup, we were able to demonstrate the superiority of the second image description (Chapter 4). In theory, we would like to test it against all types of objects, pictured under all possible

imaging conditions, a test which is practically impossible. In practice, we do not expect it to work on all objects; instead, we follow a more pragmatic approach. We let the users decide whether an automatically generated structure is appropriate and if otherwise, offer the option to change it accordingly. This is where the image hierarchy editor serves its purpose.

Another reason to introduce user interaction lies with that of personal preference. It is widely known that the problem of visual perception is self-dependent [71], i.e., we interpret objects in rather different ways. Depending on who is looking, an object might have more than one valid structure. Take a commercial drinks-bottle as an example, one might break it into a cap and a main body; another might be much more detailed and produce a cap, bottleneck, mid-body and lower-body decomposition. This is particularly important when we seek to make artworks from object structures later in the thesis, simply because art itself is a highly subjective thing.

5.2 Difficulties Behind the Editor

We would like an editor that can be used to alter automatically generated structures in a simple and efficient way. As a result, rather than building an object hierarchy from scratch, one can be obtained through a few mouse clicks.

Nevertheless, editing hierarchical image descriptions is a challenging task. Challenges mainly come from the degree of complexity that is intrinsic to the hierarchy itself. Figure 5-1 and 5-2 provide two example image hierarchies generated using hierarchical agglomerative clustering (Chapter 4). On the left of each figure, image primitives are shown; while the final hierarchy is provided on the right. As can be seen, automatically generated image hierarchies are rather large and complex. The "bison" image shown on the left of Figure 5-1 is relatively simple in the sense that the background is plain and the "bison" as foreground is of a rather different colour than the background. Even for such a relatively simple image, the corresponding hierarchy can still be complex. This particular image hierarchy has 311 leaf nodes (image primitives), 621 nodes in total and 18 levels. In general, because of the binary-split nature of such type of image hierarchies, we can expect $2N - 1$ number of nodes for each, where N is the number of image primitives.

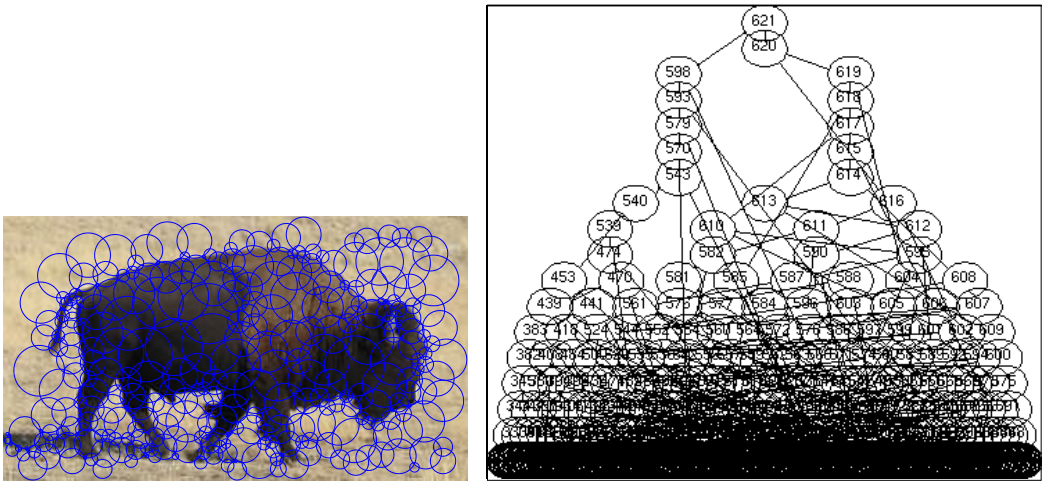


Figure 5-1: Left: region primitives overlaid on the original image; right: the corresponding image hierarchy.

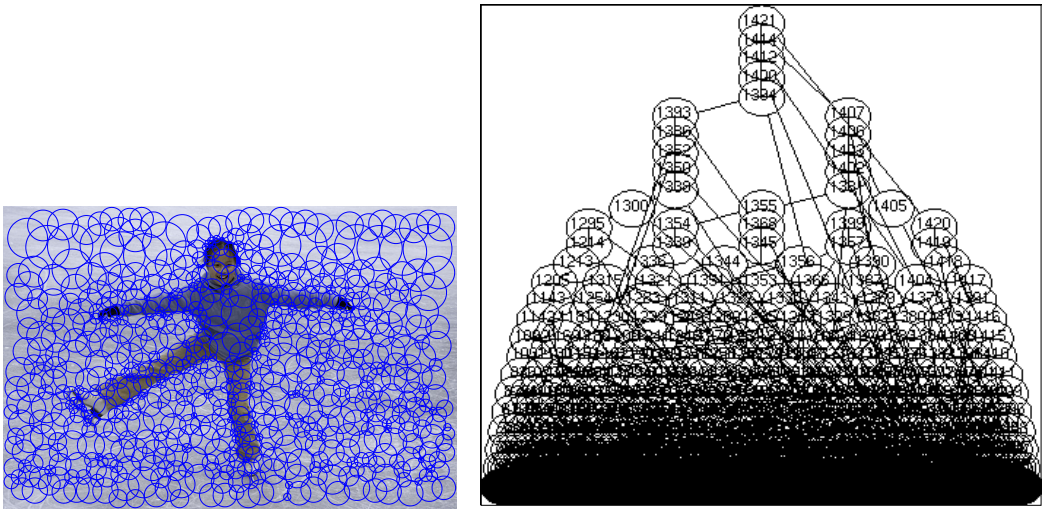


Figure 5-2: Left: region primitives overlaid on the original image; right: the corresponding image hierarchy.

In terms of image content and colour variance, "skater" of Figure 5-2 is not too much more complicated, but the resulting hierarchy still has 711 leaf nodes, double that of the "bison" hierarchy. We can make use an image of a natural scene, such as the one shown on the left of Figure 5-3, as an example to demonstrate how complicated automatically generated image hierarchies can get. The resulting hierarchy of that image contains an astonishing 5126 leaf nodes, 10251 nodes in total which spread across 67 levels. It is so huge that any meaningful visualisation is impractical.

Having just observed the degree of complexity associated with image hierarchies

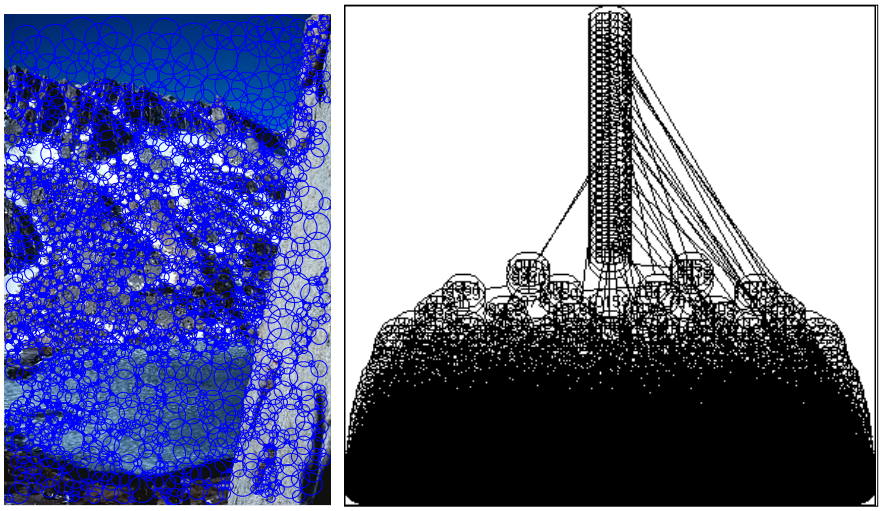


Figure 5-3: Left: region primitives overlaid on the original image; right: the corresponding image hierarchy get too large to have any feasible visualisation.

generated by agglomerative clustering, it might seem that introducing user interaction is not feasible. In the next secion, we will describe how graph energy can be used to ease the editing process.

5.3 Using Graph Energy to Assist the Editing Process

As previously mentioned, we would like to integrate user interaction into the process of making image hierarchies, so that: (i) false hierarchies can be amended; (ii) user preference is kept, as people tend to interpret objects in different ways. However, from the last section, one might gather that image hierarchies can be too complex to interact with or even infeasible to display at times (Figure 5-3). In this section, we will demonstrate how graph energy enables an easy and intuitive editing process.

Graph energy was introduced in Section 4.3.2 as an automatic mechanism to determine salient segmentations of images, where each segment is a node in the corresponding image description. Alternatively, it can also be treated as means of filtering the image hierarchies, so to identify salient nodes. In exactly the same way, we use graph energy in the editing process to propose potential nodes of interest, which are presented to the user as image segments. By doing this, we

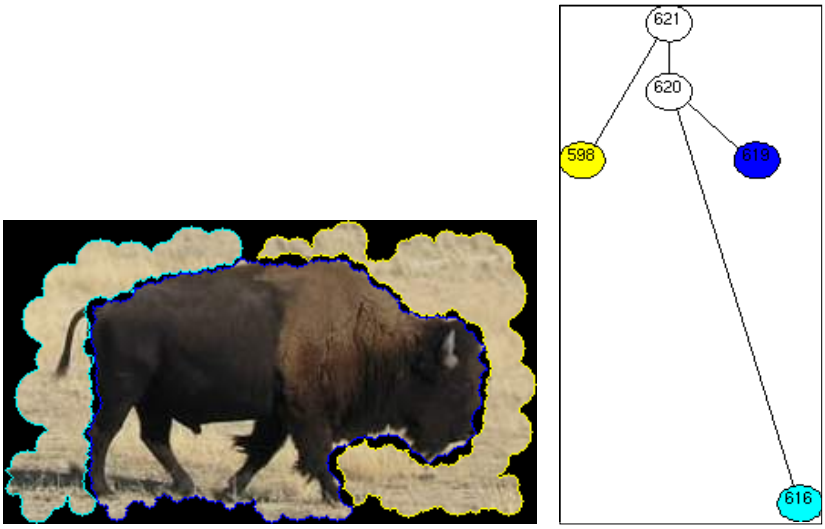


Figure 5-4: Left: automatically generated structure; right: graph representation of the structure.

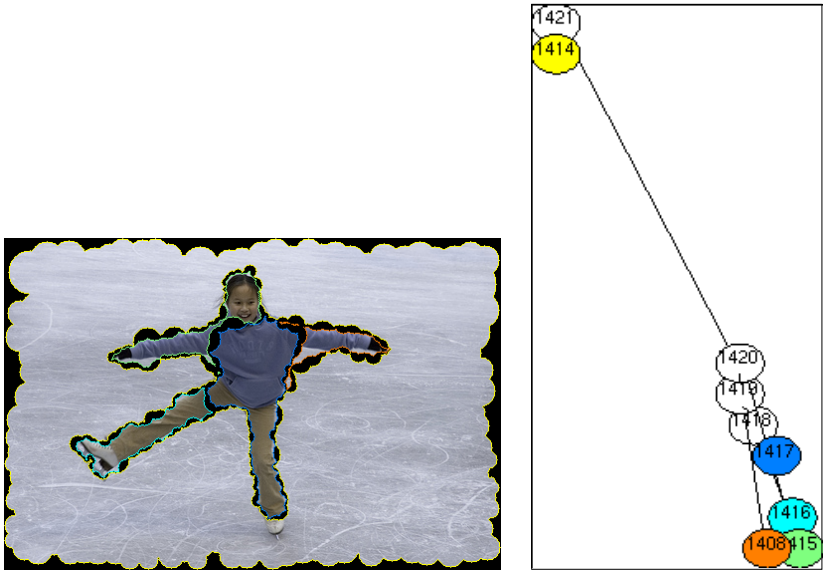


Figure 5-5: Left: automatically generated structure; right: graph representation of the structure.

reduce the large image hierarchies to something that the user can easily interact with. Moreover, these salient nodes are often quite close to the desired ones, so that little editing effort is required.

To understand this concept better, results of applying graph energy to the image hierarchies shown in Figure 5-1 and 5-2 are provided in Figure 5-4 and 5-5, respectively. In both figures, image segments and their corresponding nodes in the filtered hierarchies are colour-coded. It can be seen in the "bison" example

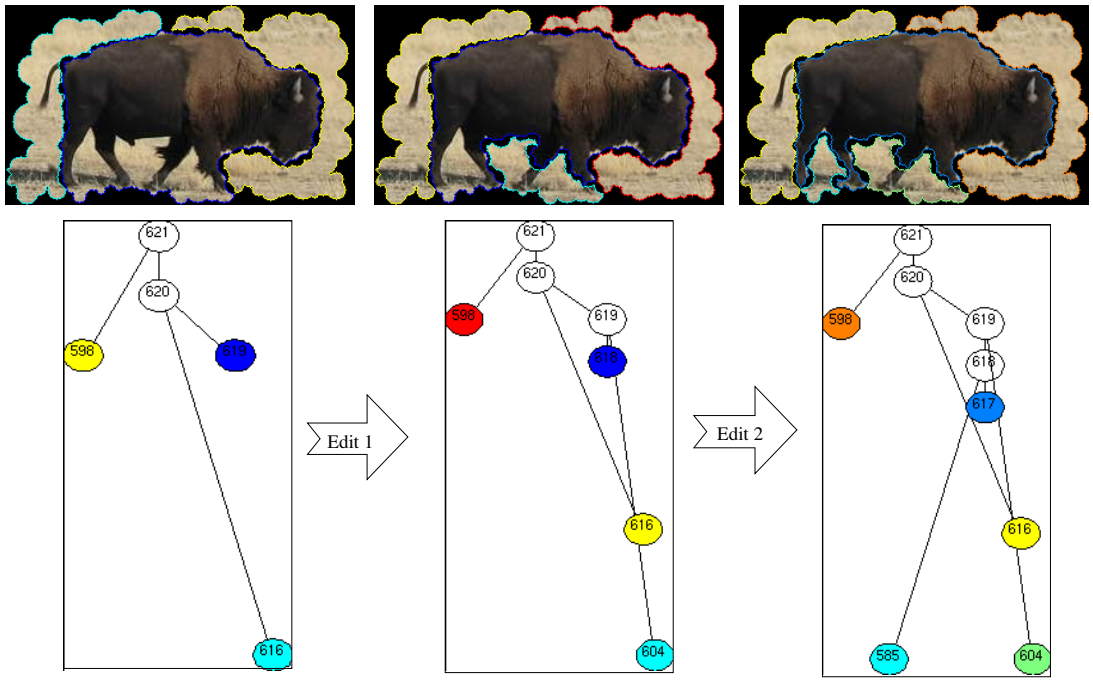


Figure 5-6: Looking at the underlying structure below the node corresponding to the "bison". From left to right: the automatically segmented "bison"; breaking the "bison" node into its children; following the children of the refined "bison", which leads to the true segmentation of it. Note: the user clicks on the actual image to edit the hierarchy, rather than the hierarchy itself.

(Figure 5-4), graph energy managed to automatically break the image into three parts, two of which are the background and the other corresponds to the "bison" itself. The filtered image hierarchies are shown on the right of both figures. It is important to note that the automatically generated hierarchy of 1421 nodes is now reduced to a very simple graph consisting of 8 nodes. A similar trend can also be observed in the "skater" example (Figure 5-5).

We should emphasise here that although large sections of the original hierarchy are being filtered, the original hierarchy itself is kept. Each node in the filtered hierarchy still has a rich underlying structure and can be brought to use when needed. This underlying structure is important to the overall editing process, as will become clear in the following section, where the design of the editor is discussed. Yet, in order to have an early grip on the concept, let us have a closer look at the structure underlying the node (node 619) corresponding to the "bison" in Figure 5-4. One might observe that the "bison" is actually not perfectly segmented out, in the sense that areas between its legs should not count as parts of the "bison". What happened in this particular case is

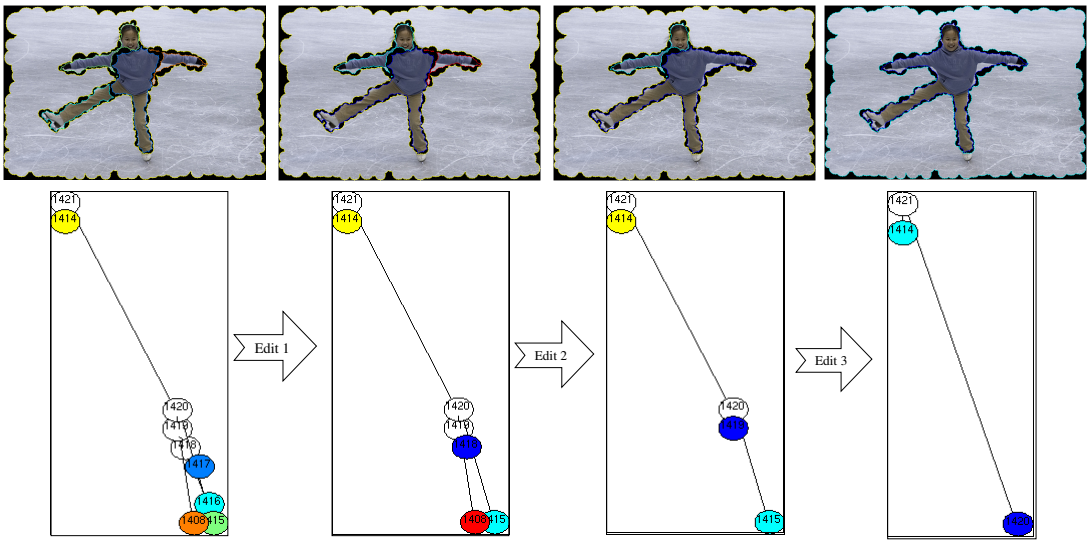


Figure 5-7: Demonstrates how regions are merged by following the parent nodes in the hierarchy. From left to right: the automatically segmented "skater"; merging the right leg to the body; merging the leg arm to the refined body; one more merging operation leads to the true segmentation of the "skater". Note: the user clicks on the actual image to edit the hierarchy, rather than the hierarchy itself.

that graph energy somehow finds the "bison" with those areas in-between its legs more stable than the true "bison". Such a decision is also to do with how graph energy is used. As mentioned in Section 4.3.2, we are currently using the last local minima among graph energy values calculated throughout the merging process. This implies that the segmentation produced using graph energy tends to be "conservative", i.e., graph energy leaves broad image segments whenever possible. So it would not be surprising to observe that the bison is slightly under-segmented. However, as demonstrated in Figure 5-6, little effort is required to obtain the true "bison". In that figure, we show what happens when we follow the parent-children relationship on the "bison" node; by following just two levels down this node, the true "bison" can be obtained (node 617).

Different from the under-segmented "bison", "skater" in Figure 5-5 is rather over-segmented as a whole. However, its parts are still under-segmented; for example, it might be desired for the "skater"'s head and right arm to be separated, and the same goes for her body and left leg. If we choose to extract "skater", instead of splitting nodes, we travel up the image hierarchy. At each step, we merge two children nodes into a single parent node, whose corresponding region is just the union of the regions of the two children nodes. This process is illustrated in Figure 5-7. We only had to perform three merging operations to get a nice

"skater" and skating ground separation.

To summarise, automatically generated image descriptions are often too complicated to be edited. However, using graph energy we are able to filter out more meaningful hierarchies, which are often a few edits away from the desired ones. Figure 5-6 and 5-7 provide two examples showing how well graph energy filters automatically generated image descriptions. In both examples, we only had to perform two/three edits in order to reach the desired outcome. Here, an edit refers to the operation to split a parent node into its children nodes by going down the hierarchy, or merging two children into a single parent node by going up the hierarchy.

5.4 Designing the Editor

Previously, we analysed the degree of difficulty the editing task entails and demonstrated how graph energy can be used to boost editing efficiency and effectiveness. In this section, we will detail the design of the editor and demonstrate how the editing process can be integrated into a single GUI.

The editing process can be broken down into two major steps:

First step — Object Identification: Users are asked to select an object in the image that they want the structure of by clicking mouse buttons over it.

Second step — Structure Extraction: Users will be able to apply graph energy to the selected object, in a recursive fashion, and refine the proposed decomposition when needed.

During the first step, objects are selected by clicking on a segmented image automatically produced using graph energy, such as the images on the left of Figure 5-4 and 5-5. We found that in most cases, graph energy is able to distinguish between background and foreground regions. Therefore, identifying the foreground object often involves only one mouse click. In addition, the user also has the option to refine the proposed segmentation by merging and splitting regions. For instance, the user needed to perform two splits in order to get the "bison" (see Figure 5-6).

After the object selection step, we have a collection of image primitives corresponding to the selected object, which is highlighted in red. The user is then able to apply graph energy to break the object into parts and also refine the decomposition by splitting and merging regions.

In a few special cases, the object that the user wants might not already exist in the hierarchical description. In other words, there are no nodes in the hierarchy that correspond to the desired object. The editor should therefore offer the ability to create new nodes by merging and splitting existing nodes, so that the desired object can be found. For instance, the left shoe of the skater in Figure 5-7 is associated to the background, in order to include it with the skater the user needs to perform a merging operation (illustrated in Figure 5-8). However, it is important to note here that by creating new nodes and recursively applying graph energy on them, new hierarchical descriptions will be generated. The resulting new description will be completely independent of the original description computed from the entire image. The newly formed descriptions will however share a subset of image primitives used in the original descriptions.

Apart from assisting the process of finding object parts, another important function of the editor is establishing relationships between these parts. The topological structures of objects exist amongst its parts, yet, such relationships are not recorded during the editing process.

In summary, the hierarchy editor can be seen as consisting of two steps: an object selection step where the user selects one or more objects he/she wants and a structure generation and editing step where the user gets to edit the proposed structure of an object. Both steps will be described in detail in the next section, where we talk through a typical editing process.

5.4.1 Using the Editor: A Walk-through

In this section, we demonstrate the hierarchy editor by walking through the editing process with the "skater" image shown on the left of Figure 5-2 as an example. The controls of the editor are simple and intuitive, which will become apparent in the following walk-through.

Figure 5-8 illustrates a typical editing process:

Step 1 (Figure 5-8(1)): As a start, the user is presented with a segmented image, obtained using graph energy; the corresponding filtered image hierarchy is shown as well. Each partition of the image is colour-coded. Ideally, the object that the user wants would have already been segmented out, i.e., the object resides in one and only one partition. In this case we can then select it using a single mouse click. However, in some cases, the user would still have to make a few edits to get the object exactly as he/she wants. In the example shown, the user wants to select the "skater", which is rather over segmented.

Step 2 (Figure 5-8(2)): The user tries to merge the segmented parts of the "skater" by going up the original image hierarchy, a process also demonstrated in Figure 5-7. This process involved **3 left mouse clicks**. The merging result is shown.

Step 3 (Figure 5-8(3)): A skating shoe belongs to the background segment, the user wants to include it with the "skater". In order to do this, the user has to break the background region. Again, the image hierarchy is proven to be helpful here; the user only had to go two levels down the background node to find the node corresponding to the skating shoe. This process took **2 right mouse clicks** in total.

Step 4 (Figure 5-8(4)): In order to merge the old "skater" and the missing shoe, the user needs to place markers on each by left clicking while pressing down the "shift" button (Figure 5-8(3)); afterwards, one needs to press the middle mouse button to group. The new "skater" is now complete and is highlighted in red (Figure 5-8(4)). The user is able to select another object by left clicking on it after pressing the "Select" button. This process involved **5 mouse clicks in total**.

Step 5 (Figure 5-8(5)): The result of breaking the selected object (highlighted in red in Figure 5-8(4)) using graph energy. This is done by pressing the "Break" button. This screenshot shows the result of applying graph energy on the "skater". Breaking an object only takes **1 mouse click** on the "Break" button.

Step 6 (Figure 5-8(6)): As shown in Figure 5-8(5)), graph energy managed to break the "skater" into 4 different parts, viz, head, left leg, right leg and

body with arms. The user can then refine the proposed parts by merging/splitting using left/right clicks, respectively. At this stage, the user has the option to label each part of the object. This screenshot shows the user split the node corresponding to body with arms, into body, left hand, right hand, left arm and right arm; and labelled other nodes accordingly as well. A graph visualisation is also shown to the right of the editing panel. Such detail editing took a total of **17 mouse clicks**, **8** out of which was spent on naming parts.

Step 7 (Figure 5-8(7)): The user is able to break parts of the object further. For example, here, he/she further split both legs of the "skater" to account for skating shoes separately. The resulting hierarchy can be seen on the right. This process took a total of **9 mouse clicks**, **4** out of which was spent on naming parts.

Step 8 (Figure 5-8(8)): To now, the editing process is finished. Yet, the user has the option to visualise the toplogical object structure by overlaying it on top of the decomposed object. This is done by menu selection.

Till now, the user has successfully extracted the structure of the skater through only 37 mouse clicks. It is worth noting that despite the simple nature of the structure, each part still has a rich underlying image description underneath which becomes useful if any further editing is needed.

5.5 Conclusions

In this chapter, we have implemented a manual editor for users to interact with the automatically generated image hierarchies. We have shown that the editing process is often quite painless, which usually requires a few mouse clicks to reach the desired outcome. Such efficiency is made possible because of the rich underlying image descriptions and more importantly, because of a graph theoretic measure to automatically break objects into its parts.

In the rest of this thesis, we will further demonstrate the value of the proposed image hierarchy editor by utilising it in the application of synthesising abstract artworks from photographs (Chapter 7).

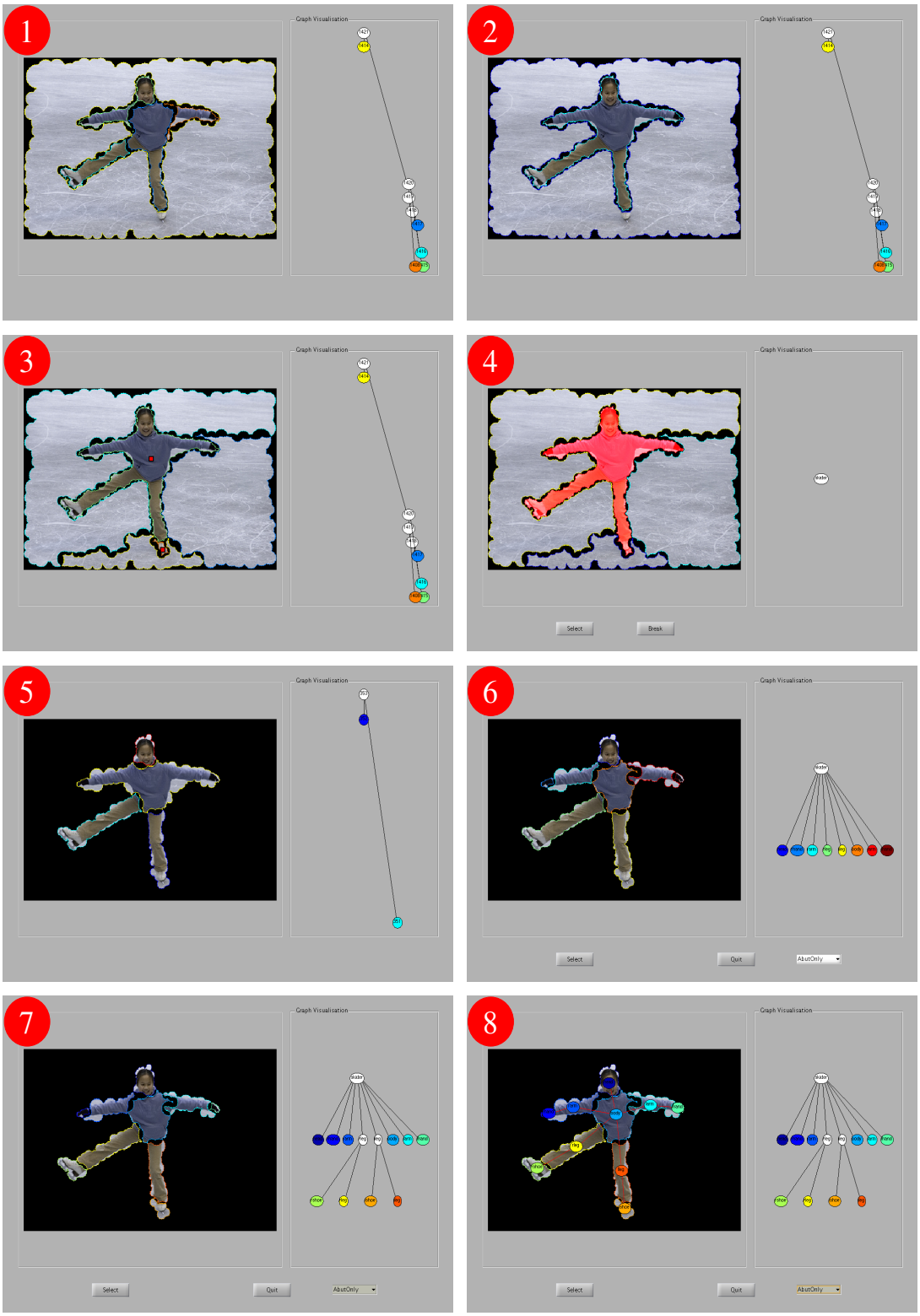


Figure 5-8: A typical editing process of the "skater". The user interacts with the hierarchy by clicking mouse buttons on the image.

Part III

Classification and Painting Using Hierarchical Image Descriptions

Chapter 6

Image Description For Object Classification: Structure is Invariant to Depictive Style

In this chapter, we study the problem of automatic classification of objects of different depictive styles such as photographs, paintings and drawings. By doing so, we justify our claim that topological object structures are invariant across depictive styles. Moreover, towards the end of this chapter, we will move on to demonstrate how qualitative shape labels enable matching on a part-to-part basis. Shape labels are obtained by optimally fitting simple geometric shapes to object parts.

There are two main parts to this chapter, each occupying a section. In Section 6.1, we explain how object classes can be automatically learnt from previously proposed hierarchical descriptions. The significant novelty is the use of fixed length feature vectors to encode structures. This enables us to form object classes using images of different kinds: photographs, paintings, and drawings. Afterwards, in Section 6.2, we move on to study how fitting simple geometric shapes to object parts helps to extend the object classification framework to enable matching objects across different depictive styles. In the same section, a novel automatic shape selector that is able to choose the "best" shape amongst a few is proposed. The chapter is then concluded in Section 6.3.

6.1 Classifying Objects of Different Depictive Styles

In this section [1], we study the problem of classifying objects of different depictive styles. In particular, we test our proposition that topological object structure is an invariant property across depictions.

From an input image we can already build a hierarchical image description and automatically extract objects and their parts. The previous part of the thesis (Part II) includes detailed explanations of both hierarchical image descriptions and the object extraction process. In this section, we will first briefly review the automatic object structure extraction process (Section 6.1.1). Afterwards we will move on to explain how graph representations of the automatically extracted structures can be encoded into a feature vector of fixed length and how such feature vectors can be used to classify (Section 6.1.2).

6.1.1 Object and Object Part Identification

The first step is to extract topological object structures from images. As previously demonstrated in Part II of this thesis, graph energy offers a promising way to break an object into its parts. We also use graph energy here to identify objects and their parts, in a recursive fashion.

Given an image, at its first application, graph energy breaks the image into the fewest number of parts, which tend to be of large scale and correspond to coherent regions such as "background" and "foreground". There is no way, as yet, to automatically differentiate foreground from background; we solve this by the user clicking a mouse to indicate the object. This is the only user interaction in the whole process. We might eliminate it using a heuristic (e.g. objects cover the central region of the image) but the cost will be an assumption about image content. We designate as "objects" the subtrees that have just been identified as foreground. To extract its parts, we recursively apply graph energy to this object (subtree). We are able to apply graph energy easily to any object and any part, which breaks them into their constituent parts.

[1] This section contains joint work with Xiao Bai, Department of Computer Science, University of Bath.

The structural relationship between an object and its parts is inherited from the hierarchical relationship in the image description. The images in Figure 6-1 show how our algorithm can break images into objects, and objects into parts. It does so over a wide range of depictive styles and automatically (excepting a mouse click to identify foreground from background). Notice that although the parts vary over a visual object class, that inter-class variation seem to the eye to be wider than intra-class variation. For example, faces have several internal parts whereas flowers have one, while horses have many external parts. The structural relationship we obtain between an object and its parts reflects that in the underlying objects. Horses do have legs, a body and a head; faces have eyes and mouths; and flowers have petals and a centre. Thus we have automatically constructed a description which is indicative of the structure of real world objects. This structure can be treated as a graph of nodes and arcs, where nodes correspond to object parts and arcs are links inherited from the original image description. In this next section, we will demonstrate how such graphs can be clustered in feature space.

Figure 6-1: Examples objects and their parts, found using graph energy analysis, taken from our experimental training and test sets. Facial parts are identified; four legged animals have their head and legs separated from their body; flowers have their centres extracted.

6.1.2 Structure Vectors as Object Features, and Classification

The graph of an object and its parts is a structural feature. Nodes are labelled with the statistical distribution of measures that characterise a region, in this case colour and gradient magnitude. We could use other characterising measures, such as SIFT features for regions [89], relative location or indeed any other thing we find convenient. However, we choose to progress instead by considering the structure we have just constructed, and only that. This is (i) to see how useful structure alone is as a classifier; (ii) to create equivalence classes sufficiently broad to encompass many depictive styles. Once we begin using node labels it is easy to unwittingly make tacit assumptions about content, assumptions we wish to avoid.

We now assume we have a collection of unlabelled, undirected graphs G_i. Each one corresponds to an object in an image, and is typically small because the number of any object is usually composed from only a few parts. We use graph spectral theory to compute a vector of fixed size for each one. This vector we call a structural feature vector, because it encodes structure and is a feature of an object. It can be treated just as any other feature vector. There any many ways to create a structural feature vector from a graph. We follow Zhu and Wilson [186], who advocate the use of eigenvalues from the signless Laplacian matrix because of its stability to structural variance. If A is the adjacency matrix for a graph and D is the degree matrix, the signless Laplacian is $D + A$. The eigendecompsition of the signless Laplacian is $UCU^T = (D + A)$, and the feature vectors we use are the first eight eigenvalues of $F = diag(C)$; if F has fewer than eight elements we pack with zeros. This is equivalent to adding isolated nodes to the graph.

Structural feature vectors have the distinct advantage that they can be clustered — it was this property that motivated their use. We built a classifier using the training examples in Figure 6-2. It contains thirteen different classes, each a column in the figure. Each class has seven exemplars; this is far fewer is typically used to build visual object, for example classifiers [81, 51, 96], and compares very well with the "one shot" learning of Fei-Fei et al. [37]. Notice that each class contains examples in several depictive styles.

We used a semi-supervised approach when building a classifier, as is common.

Figure 6-2: The set of training images. Each column is a class (but horses and cows are both four legged animals). Many depictive styles are included in each class.

This method was chosen because examining a scatter plot of the largest eigenvectors suggested some categories are too close to separate automatically. We manually labelled images showing objects of a single visual class. Each class has a single Gaussian distribution fitted to the structural feature vectors. The Gaussians were then assembled into a single mixture model, which is seen in Figure 6-3. All sets (Gaussian components) contains drawings and paintings as well as photographs, yet form discernible clusters. The spread of the cluster is explained by noise (missing or additional nodes and arcs) that can arise from slightly different segmentations.

One can observe twelve distinct clusters in Figure 6-3, yet as stated we used thirteen visual classes. The explanation is that cows and horses have fallen into the same equivalence class. This is not a confusion but instead is a consequence of the broad categories that form when structure is used. Cows and horses share a common structure and the classifier reflects this. This grouping could be useful — after all, a user of some content based retrieval system might want "four legged animals". The implication is that horses and cows must be separated by some feature other than structure.

To verify our classifier we used a set of test images, with 50 images from each class, some of which are seen in Figure 6-4. Test images were highly varied in their depictive style. For a given test object we constructed a feature vector,

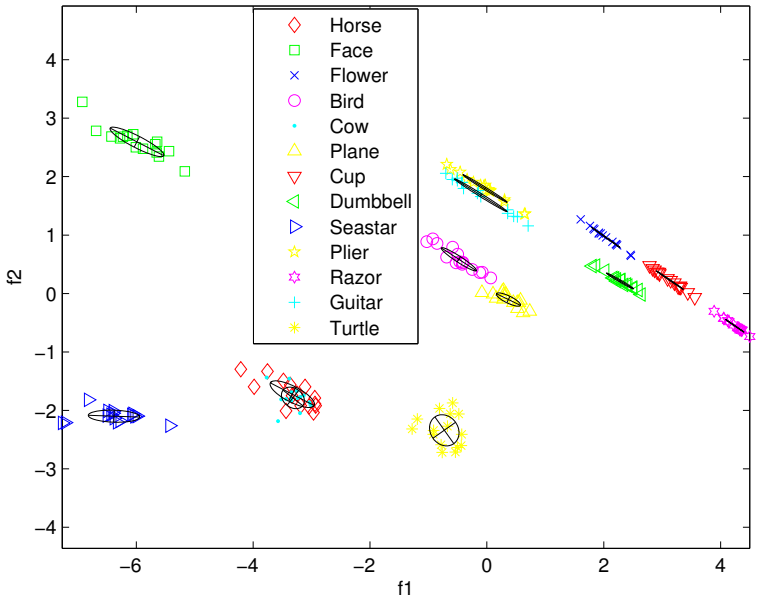


Figure 6-3: Classes as components in a Gaussian Mixture Model. Note: only the first two dimensions of the feature vector is shown.

Figure 6-4: A subset of testing images. Each column is a class (but horses and cows are both four legged animals). Many depictive styles are included in each class.

which was then input to our GMM which estimated the posterior probability of the test object belonging to each training class. The test object was assigned the class for which the posterior probability was largest. We constructed a confusion matrix of how the test set distributes over the training classes, which can be seen in Table 6.1.2. It is clear that most objects can be correctly classified with high probability, suggesting our GMM is a reasonable discriminative model.

Despite classifying many images correctly, we of course meet with some failures.

	1	2	3	4	5	6	7	8	9	10	11	12
1	0.94								0.03			0.03
2		0.94		0.06					0.03			0.03
3			0.86				0.14					
4		0.06		0.91		0.03						
5					0.86	0.14						
6			0.06			0.89					0.05	
7							1.00					
8	0.06							0.94				
9									0.97		0.03	
10						0.06				0.94		
11							0.09			0.03	0.88	
12	0.03											0.97

Table 6.1: A confusion matrix; each row shows the probability a test class is classified as a given training class. The classes are: (1) Four legged animals; (2) Faces; (3) Flowers; (4) Birds; (5) Plane; (6) Cup; (7) Dumbbells; (8) Star fish; (9) Pliers; (10) Razors; (11) Guitars; (12) Turtles.

In Figure 6-5 we see some faces that have been incorrectly classified. We notice that the incorrectly classified faces fell into categories which are neighboring to the face class (see Figure 6-3). We have yet to observe incorrectly classified faces falling further afield.

Figure 6-5: Examples of incorrectly classified faces. Left to right the faces were classified as starfish, horse, and bird.

To summarise, this section tests our proposition that structure can be used to classify, just as other features like texture, shape are used. The characteristic that structure brings into classification is that of broad classes — sufficiently broad to cover objects depicted in a wide range of styles: photographs, drawings, and so on, yet sufficiently discriminative to be meaningful.

We have provided experimental evidence in support of this claim. By using feature vectors to encode structure in a vector of fixed length, we have been able

to build a classifier that successfully recognises novel inputs. The broadness of the classes we generate are illustrated not just by the inclusion of many depictive styles, but also by the fact cows and horses classify as one. As mentioned already, there will be applications and contexts when this is desirable, otherwise intra-class discrimination will have to be on the basis of some other property.

6.2 Extending to Matching: Shape Fitting and Automatic Shape Selection

Previously in this chapter, we demonstrated how topological object structures form meaningful clusters in a structural feature space. Topological object structures offer middle-to-high level descriptions of objects. It is essentially such abstraction that enabled classification of objects from a wide range of depictive styles.

Object structures capture the relationships among parts of objects. However, during the classification process we didn't employ information carried within object parts at all. We filtered out such information on purpose - they are often dependent on particular depictions. Nonetheless, information of object parts should be considered say, when we want to match two objects. Using the previous classification framework, we are able to tell that two objects are of the same class. In theory, incorporating part information would also boost the classification performance. However, as previously mentioned, the form of such information is crucial. Standard photometric measures such as gradient intensity, colour histograms and so on, are not invariant to changes in depictive styles. We shall again seek middle-to-high level representations for each part of the object.

Following the above argument, in this section [2], we briefly study the possibly of using qualitative shape labels to augment object structures. The rationale for using shape labels is that it provides a quick and easy way to match, and is naturally invariant to many geometric transforms, to clutter, and noise: matching photographs to artwork makes all of these demands. Specifically, we assign a qualitative shape label to each node corresponding to every part of the object, with the hope that such augmented information will: (i) introduce part-to-part

[2]The shape fitter and selector are joint work with Dr. Paul Rosin from University of Cardiff; the quantitative shape selector experiment was conducted by Anupriya Bilaka.

matches; (ii) deliver a level of abstraction useful for other applications, such as NPR. In the rest of this section, we will first describe how shapes are optimally fitted to each object part (Section 6.2.1); a novel classifier is later proposed to select the best shape among a few for a given part, as discussed in Section 6.2.2.

6.2.1 Fitting Simple Shapes to Regions

In this section, we explain our approach of optimally fitting simple geometric shapes to objects and their parts. Specifically, we fit five shapes: circles, rectangles, triangles, superellipses and a "robust" version of the convex hull. It is worth noting that the proposed shape fitting techniques work on any image segment, represented as its corresponding boundary curve. A few examples of inputs to our shape fitter are shown in Figure 6-6.

Figure 6-6: Example input segments to the proposed shape fitter. From left to right: body, head and front leg of a bison.

Voss and Süße described a powerful method for fitting a variety of geometric primitives by the method of moments [169]. The data is first normalised by applying an appropriate transformation to put it into a canonical frame. The fitted geometric primitive is then simply obtained by taking the geometric primitive in the canonical frame and applying the inverse transformation to it. For instance, for an ellipse they take the unit circle as the canonical form, and apply an affine transformation consisting of a translation to set the moments $m_{10} = m_{01} = 0$ and an anisotropic scaling such that $m_{20} = m_{02} = 1$. We have applied this approach to fit ellipses, rectangles and triangles.

To fit superellipses a closed form solution is not available using the above approach, and so we use the least squares method described in [125]. The ideal distance measure to minimise would be the shortest Euclidean distances between each point and the superellipse, but this is expensive to compute. Instead the ray from the centre of the superellipse to each data point is intersected with the superellipse, and the summed distances along these rays is minimised using Pow-

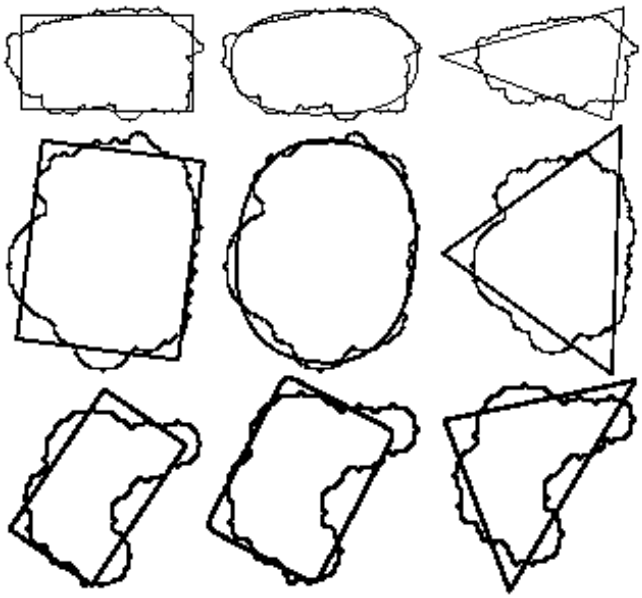


Figure 6-7: From left to right (column-wise): Examples of fitting rectangles, superellipses and triangles, respectively.

ell's method for non-linear optimisation [117]. The optimisation is initialised by fitting an ellipse.

The convex hull is an attractive symbolic representation of a shape on two counts. It is generally more compact (using only a subset of the original polygonal vertices), and also perceptually simpler since all indentations have been removed. However it has two limitations: it is insensitive to the size and shape of all indentations, and is also too sensitive to protrusions. To overcome these problems Rosin and Mumford [124] suggested a "robust" version of the convex hull, which is the convex polygon that maximises the area overlap with the input polygon. To compute the robust convex hull they used a genetic algorithm; alternatively a dynamic programming solution has been described [75].

Results of fitting various user defined shapes to the input segments shown in Figure 6-6 are given in Figure 6-7.

6.2.2 Automatic Shape Type Selection

We are now able to optimally fit a collection of simple shapes to a given region. The problem now is how to choose amongst them. Interaction is one approach, but not only is this tedious for the user but, we argue, is less interesting than

considering automatic choice.

Others have approached automatic selection through an information theoretic measure of some kind; Gheissari and Bab-Hadiashar [49] provide a review and an empirical comparison of these. One of the earliest being Akaike's information criterion (AIC) [1]. However, due to the different assumptions made by the various criteria regarding the distribution of the data, the different measures can give quite different results. For instance, Schwarz's Bayesian information criterion (BIC) [138], which is similar to the Minimum Description Length (MDL) criterion, penalises free parameters more strongly than AIC. Another criterion, the "geometric information criterion", was introduced by Torr [161] and later the "geometric AIC" was suggested Kanatani [70]; both of these were specifically designed for computer vision applications. It is clear there is no single agreed way to fit some shape from more than one family. Issues of concern in the mathematical and computer vision literature are the robustness of the fit with respect to outlying data points, and the invariance of the fit under transformations of the data. The most common types of fitting in computer vision minimise some function (e.g. sum of squares) of the residuals. We note that measures are usually chosen for their mathematical tractability and computational convenience and complexity, rather than how well they correspond to perceptual or aesthetic judgements. Yet if we are to match photographs to human-made artwork, and later to process a photograph into a synthetic artwork these value judgements are crucial. We have therefore opted to use a classifier which is trained under human supervision, in the hope to retain some degree of subjectivity. We now explain our classifier and the training regime.

Automatically selecting appropriate shape models is done using a supervised classification paradigm; specifically, a C4.5 decision tree [120] is learnt from a training set of regions which is then applied to new unseen data. C4.5 decision tree benefits from being able to deliver a relatively decent classifier with minimal training data. Other machine learning techniques such as support vector machines and neural networks can be used here, however, we found C4.5 trees sufficient for the purpose of this work. The basis of a decision tree is that each feature can be used to make a decision that splits the data into smaller subsets, partitioning feature space into equivalence classes using axis-parallel hyperplanes. C4.5 builds decision trees by selecting the most informative feature (that is not yet considered in the path from the root) to split each subset. An entropy measure — Normalised

Information Gain — determines the effectiveness of each feature. The regions are described by a feature vector and are manually labelled into shape categories. These features are the basis for making the decision regarding which is the most appropriate model. The feature vector consists of the errors between the region and each of the fitted shape models. To compute the errors at each data point the shortest distance to the fitted shape is determined using the distance transform. However, the summed error is not a sufficient descriptor – it is easy to construct examples for which the best shape model (according to aesthetics and perceptual criteria) does not have a lower summed error. Instead, the distribution of point errors, which is more informative, is considered, and summarised by the following lower order statistics: mean, standard deviation, skew, and kurtosis. Thus, for five shape models and four statistical terms, each region is described by a total of twenty features.

Following this approach, we can easily train an automatic shape selector. However, an obvious question would be when should the training stop, i.e., how much training data is really needed to obtain to decent shape selector? We will go on to answer this question in the next section.

Figure 6-8: Two images used to train the automatic shape selector

In practice we selected two training images other than the ones used for testing purposes, both illustrated in Figure 6-8. Indeed, we can select any set of images, provided that we can get a relatively sufficient amount of training shapes. To obtain input shapes to the classifier, we simply segmented each training image using a popular segmentation technique [24]. We also deliberately segmented

each image into different granularities to get more regions with a larger variety of shapes. Amongst over 500 segmented regions we extracted their feature vectors and manually labelled 81 of them as our training data. A C4.5 decision tree is then built using those 81 pieces of training data. The learnt decision tree is used to generate all results in the rest of this thesis. We understand that the amount of training data is relatively few, but the trained classifier seems appropriate on the few images used in this work. In the next section, we conduct an experiment to study the affect of increasing training data to the performance of the final classifier.

6.2.3 Quantitative Evaluation of Automatic Shape Selector

In this section we will first provide a simple quantitative experiment to judge the performance of the proposed shape selector; followed by answering the question set out at the end of last section, that is, how much training data is really sufficient. For simplicity, we restricted ourselves to training with the simple shapes, "Superellipse", "Rectangle", "Triangle", and (robust) "Convex hull". In order to obtain both training and testing regions, we used an off-the-self image segmentor by Cour et al. [24]. A total number of 140 shapes were used to train, 35 instances from each shape type. The confusion matrix is provided in Table 6.2.

	superellipse	rectangle	triangle	convex hull
superellipse	32	1	1	1
rectangle	4	29	1	1
triangle	3	1	30	1
convex hull	1	1	1	32

Table 6.2: The confusion Matrix, scaled by the number of instances per known class. Each row (capital letters) shows the result of classifying a ground truth set of a known shape. The fraction of times a ground truth shape class is classified as some nominated shape class is given. Each ground truth class contained N instances.

Having just evaluated the performance of the automatic shape selection technique, we continue to answer the question "when can training cease?". Related to this is "how much confidence can one have in the confusion matrix?". There is a common answer to these: cease training when the confidence matrix converges to a stable solution, so that one can have confidence it is "correct". Figure 6-9

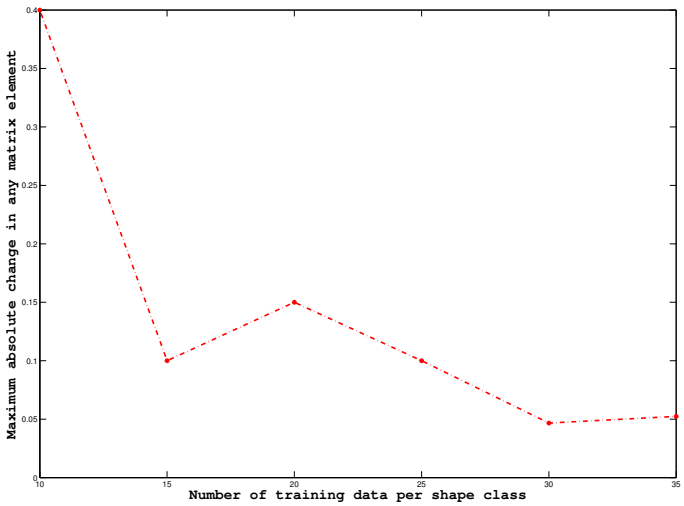


Figure 6-9: The maximum absolute change in any normalised confusion matrix element as a function of the number of training data in each shape class.

shows how the maximum absolute change in any confusion matrix element (normalised by the total number of samples at each step) depends on the number of training data. It shows we can cease training after 35 or so training data per shape class.

Until now, we can automatically assign a given image region with a shape label. Such labels can eventually be treated as abstract representations of their corresponding regions. In the next section, we will study how such shape labels alone can help to enable matching across objects of different depictive styles.

6.2.4 Matching Photographs with Artwork

We can now automatically assign shape labels to image regions. In this section, we demonstrate that it is possible to use those labels to match objects from different depictions [3]. In particular, we propose a simple matcher that matches photographs with artwork. The problem with matching in this case is that the "character" of the two images can differ significantly. Paintings can comprise large regions of flat colour, photographs usually have far more detail in them than is necessary to convey the content, including complex light effects, textures and so

[3]Work in this section was developed by Anupriya Bilaka.

on. This problem has been sparsely addressed in the literature. Schechtman and Irani [141] assert that any textures can match, provided the are self-consistent. Bai *et al.* argue that structure is a class invariant [2]. Fidler and Leonardis learn tree structures premised upon Gabor filter responses [42] and use that tree to identify many specific objects classes; some supervised training is required at the higher levels of the tree. Please note that we are only after a simple matcher here, one that is enough to demonstrate the use of shape labels in terms of extending the classification framework proposed in Section 6.1. The fusion of topological object structures and shape labels is deemed as future work.

In order to demonstrate the value of shape labels in matching, we construct a simple image hierarchy consisting of image segments of multiple scales. In particular, we use the same segmentor [24] as in shape classifier training to obtain image segments. This segmentor requires a single number, N, as input, and segments an image into that many segments. We use several values of N, specifically $N = 3i$, for $i \in [1, 8]$ to obtain 8 "levels", each of finer granularity than the last. The nodes on the different levels generate a natural hierarchy — a tree — based on overlap.

Given two images, we build their corresponding image hierarchy in the above fashion. Objects of interest are then selected manually, each of which is a subtree of the corresponding hierarchy with the object being the top node. The problem now is to find matches between two subtrees. Each node in the subtree is augmented with a shape label $\mathcal{Z}$, where $\mathcal{Z} = \{E, R, T, C\}$, corresponding to the four classes of training shape, viz, superellipses, rectangles, triangles and convex hulls. These we will call "observed" labels. The real, underlying shape label for a region is unknown to us — because the classifier may have assigned an incorrect label.

Although we have opted to use qualitative data we nonetheless benefit from a measure of the probability that two symbolic shapes match. We will estimate the probability that two observed labels a and b, correspond to the same underlying shape, the identity of which is never revealed. The confusion matrix in Table 6.2 plays a central role in this estimate. Each row of the table gives the conditional probability $p(a|Z)$, which is the probability that a known *named* shape Z which is input to the classifier is assigned the *observed* label a. For example, $p(e|T)$ is the probability that a triangle is classified as an ellipse, e. We will continue to use

upper case letters for known inputs to the classifier, and lower case for the labels it produces. Each row of the confusion matrix has constant names shape Z, each column has constant observed class a. Using Bayes' law we get the probability that a given observation a is really a named shape Z.

$$p(Z|a) = \frac{p(a|Z)p(Z)}{p(a)} \tag{6.1}$$

We know for sure that all named shapes exist at least once we assume $p(Z) = c$ for all $Z \in \mathcal{Z}$. The probability of observing the label a requires us to marginalise over the named shapes:

$$p(a) = \sum_{Z \in \mathcal{Z}} p(a|Z) \tag{6.2}$$

Now suppose we have two shapes with observed classes (i.e. a name given by the classifier) a and b. The probability that these are both of the the same named shape Z is $p(Z|a, b)$. By appeal to conditional independence (and assuming statistical independence on the observations) we get

$$p(Z|a, b) = p(Z|a)p(Z|b) \tag{6.3}$$

The probability that the observed shapes a and b are the same underlying (but never revealed to us) shape is therefore

$$p(a, b) = \sum_{Z \in \mathcal{Z}} p(Z|a, b) \tag{6.4}$$

So $p(a, b)$ is a table entry that estimates the probability that two observed labels correspond to a named shape, matching in a qualitative sense.

The probability table $p(a, b)$ is used to weight all matches between the regions in the two trees (forests). At the top-most level (the largest, coarsest regions) we consider all putative pairs of matches. We do likewise at the next level down, which expands each of the matched pairs. Recursive application generates a match-tree of all possible pair combinations. Match-tree branches are pruned where both children are not connected to both parents; i.e. if (a,b) is a parent to (c,d) and (a,c) are connected in image one, then (b,d) must be connected in image two. We seek the path of maximal probability in this tree. A path comprises of a string, $\mathcal{S}$ of matched pairs, each region appears at most once in such a string.

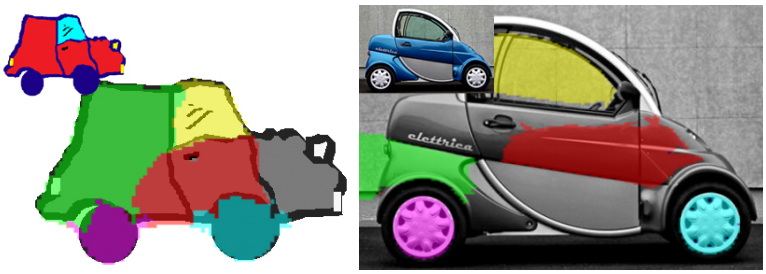


Figure 6-10: Parts of a drawn car and parts of a photographic of a car are matched, as shown by the colour coded regions.

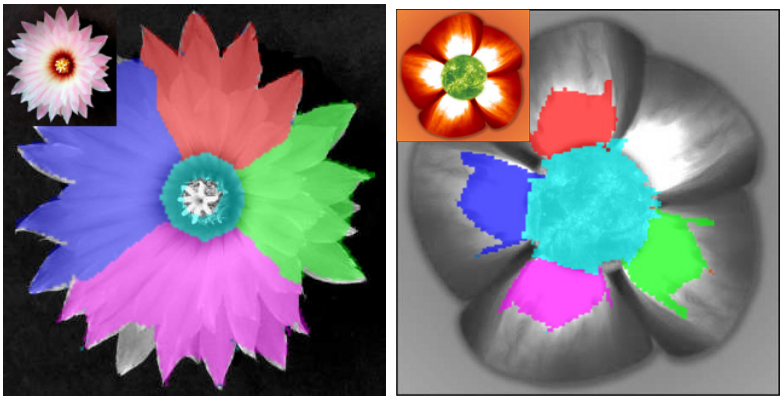


Figure 6-11: Parts of a flower are matched, as shown by colour coded regions.

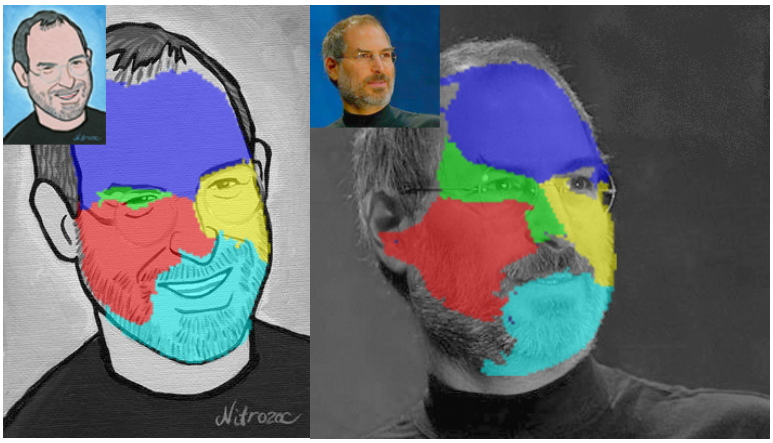


Figure 6-12: A painting and a portrait are matched, as shown by colour coded regions.

The probability of the path is then $p(\mathcal{S}) = \Pi_{(a,b)\in\mathcal{S}} p(a,b)$. Since paths can be of different lengths we normalise to take the geometric average, so $p(\mathcal{S})(1/|\mathcal{S}|)$. This is equivalent to a "characteristic" radius of a $|\mathcal{S}|$ dimensional hyper-ellipse whose main axis radii are the probabilities along the path.

Results from some of our matched photograph/painting pairs are shown in Fig-

ures 6-10, 6-11, and 6-12. The matcher has successfully matched corresponding regions in these images, even where colour and and other properties differ significantly. We are yet to test this matching scheme on more than 3 images. We suspect that it would not work equally well on a large set of images. Our explanation lies with simple image hierarchy based on the N-cut segmentor, because it can be unreliable. Despite this, these results indicate that qualitative shape can be used to as the basis of a matcher. We believe that object structures extracted from the previously proposed image descriptions would improve the performance of the matcher.

In this section, we proposed a simple matcher to demonstrate the value of quantitative shape labels in terms of matching photographs with paintings and drawings. We employed a simple image hierarchy to assist the matching process. It is important to note that the point of this matcher, though, was to explore the possibility of using nothing but qualitative shape labels to match across different depictions. Our results show that even such weak measures can prove useful. We believe that intergrating object structures with such quantitative labels is able to take us further in the direction of matching objects of different depictive styles. Nonetheless, such fusion is treated as future work.

6.3 Conclusions

This chapter argues that topological structure can be treated as a feature of an object. Consequently structure can be used to classify, just as other features like texture and shape are used. The graph spectral methods we use ensure that the classification is robust to noise. Structural classification yields broad categories such as "four legged animals". In addition, we have also demonstrated that other features such as quantitative shape labels carry the possibly of extending this classification framework to enable part-level matching.

We conclude that topological object structure is indeed an invariant property that is shared among objects depicted in different styles, i.e., photographs, paintings and drawings, which is useful for learning categories of visual objects of different depictive styles.

Chapter 7

Image Description for Image Synthesis: Generating Abstract Artworks

We have just observed how object structures can be used to classify objects in a wide range of depictive styles. In this chapter, we continue to study the use of toplogical object structures, but in the field of Non-photorealistic Rendering from Photographs (NPRP). More specifically, we will first investigate how compositions of simple geometric shapes provide abstract representations of objects and how abstract artworks of the type advocated by artists such as Kandinsky, Matisse and Melvich can be created from such representations. Later on, we will show how incorporating simple shapes with object structures yields artworks of an even more abstract nature. In particular, we are able to synthesise artworks in the styles of child drawings, cave paintings and these in the spirit of Mirö and Picasso. A single GUI which integrates all the above functionality is implemented to facilitate the art generation process.

7.1 Overview

In recent years, the NPRP literature has become increasingly populated with methods for producing abstract synthetic art. As Section 2.2 makes clear, this trend is a perfectly natural one; pioneering work relied on relatively simple image

processing to support figurative art in painterly styles such as pointillism and impressionism. Today, NPRP is capable of non figurative art, rendered in a wide range of different media. This trend of abstraction is implemented through increasingly sophisticated image understanding; high-level image understanding that facilitate the abstraction process.

Work proposed in this chapter aims to continue this trend toward abstraction. There are two crucial technical contributions that enable the creation of the kind of abstract artworks that we aim to produce, they are:

- We propose that shape simplification delivers a level of abstraction that was missing from previous attempts at producing abstract art. More specifically, as detailed previously in Section 6.2, we are able to (i) optimally fit shapes such as triangles or rectangles to arbitrary image regions; (ii) automatically decide which of these shapes is the "best" fit. We show that shapes can be rendered to emulate works of artists like Kandinsky, Melvich and later Matisse. These artists transform complex geometric shapes into much simpler forms, typically circles, squares or triangles, or else shapes resembling convex hulls, for example.

- We demonstrate that by rendering topological object structures in an appropriate fashion, artworks of an even more abstract nature can be synthesised, especially these in the styles child-like drawings, Mirö and Picasso. In particular, we show that by combining simple geometric shapes with object structures, we can go further on the road to abstraction.

In line with the above two propositions, we study the problem of creating abstract synthetic art in two stages. At first, in Section 7.2, we demonstrate how a specific style of abstract art can be automatically produced by rendering shapes alone. We then move on to investigate how topological object structures can be incorporated in order to push the level of abstraction (Section 7.3). A single GUI that incorporates shape fitting, object structure extraction and other facilitating tools in creating abstract art is also introduced in that section.

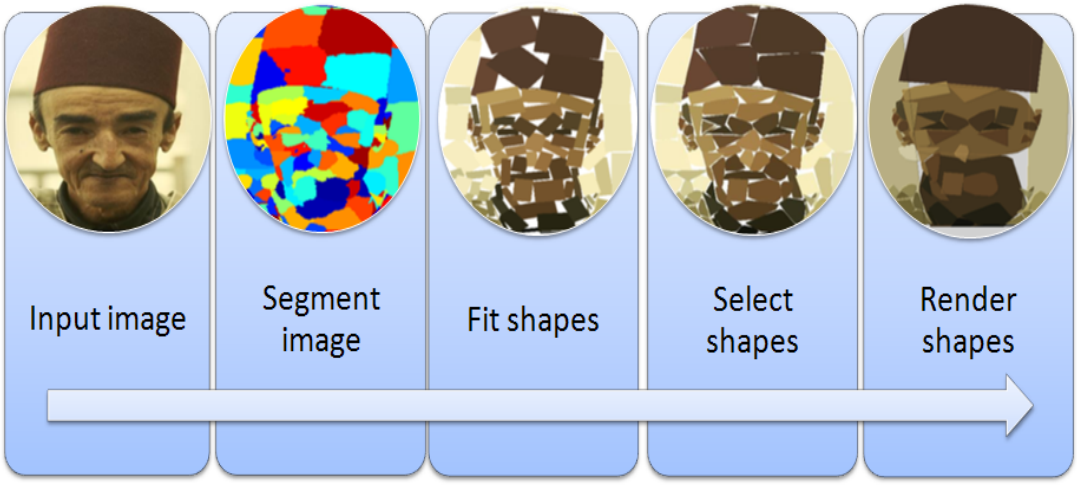


Figure 7-1: The overall pipeline of the shape rendering system. The whole process is fully automatic and only takes approx. 10 seconds from start to finish.

7.2 Generating Abstract Artwork using Simple Geometric Shapes Alone

Our proposal is that shape simplification steers NPRP away from image segmentation towards yet higher levels of abstraction. Descriptions of our shape fitting and automatic shape selection technique were already detailed in the previous chapter (Section 6.2). This section, on the other hand, describes how abstract synthetic art can be created by rendering simple geometric shapes alone. In the next section, we will continue to show how object structures extracted from the earlier proposed hierarchical image descriptions are able to deliver artworks of an even more abstract nature.

The overall pipeline of our system is illustrated in Figure 7-1. It works broadly as follows: given an image, we first build a simple hierarchy of segmented image regions, of the kind used in Section 6.2.4 of the previous chapter, but simpler. Description of this simple image hierarchy is given in Section 7.2.1. We then use the shape fitter previously detailed in Section 6.2.1 to optimally fit simple geometric shapes such as rectangles, triangles, circles, superellipses, convex hull and so on to each segment. The decision of which of these shapes is used when rendering can be specified by a user, or else can be chosen automatically (Section 6.2.2). Once shapes are fitted, we can render them, as Section 7.2.3 explains. The final piece can optionally be painted over using any one of the standard NPRP algorithms

designed for rendering areas; Section 7.3.5 provides a gallery of results.

7.2.1 Constructing a Simple Image Hierarchy

In order to test the above proposition, we built a simple image hierarchy based on an image segmented at different scales. Such a simple image hierarchy is needed because: (i) we need slightly higher-level (other then raw pixels) as image primitives to fit the shapes to; (ii) we need a simple image description that does not introduce any additional abstractness, for we aim to test the abstraction power of simple geometric shapes alone; (iii) we would also like to preserve an appropriate amount of detail given an object, segments from a low scale are often lack of detail, whereas a higher scale often results in over-segmentation, hence a hierarchical description of different scales is needed.

The hierarchy itself is merely a simple version of the one proposed in Section 6.2.4. The only difference lays with the fact that only two layers of segmentation are used, indicating foreground and background. The background layer generally consists of coarser regions, whereas the foreground layer consists of finer regions resembling object details. We will see later on in Section 7.2.3 that, such segmentation gives fitted shapes of different scales or "granularity". These different granularities are rendered so as to preserve salient detail.

A detailed description of the image hierarchy used here can be found in Section 6.2.4. However, to recap, we start by segmenting a colour image into disjoint regions of interest, using a multi-scale normalised cut algorithm by Cour et al. [24]. It has the benefit of operating in various image scales and offering a single parameter N, which is the number of regions (i.e. cuts) to make in the image, specified a prior by the user. A smaller N yields larger and coarser regions, whereas large N returns smaller and more detailed regions. Figure 7-2 shows an example colour image and its segmentation results when N is changed.

7.2.2 Fitting Simple Shapes to Regions

Having segmented an image, we are able to fit a wide selection of shapes to each region using the shape fitter described in Section 6.2.1. Specifically, we fit five shapes: circles, rectangles, triangles, superellipses and a "robust" version

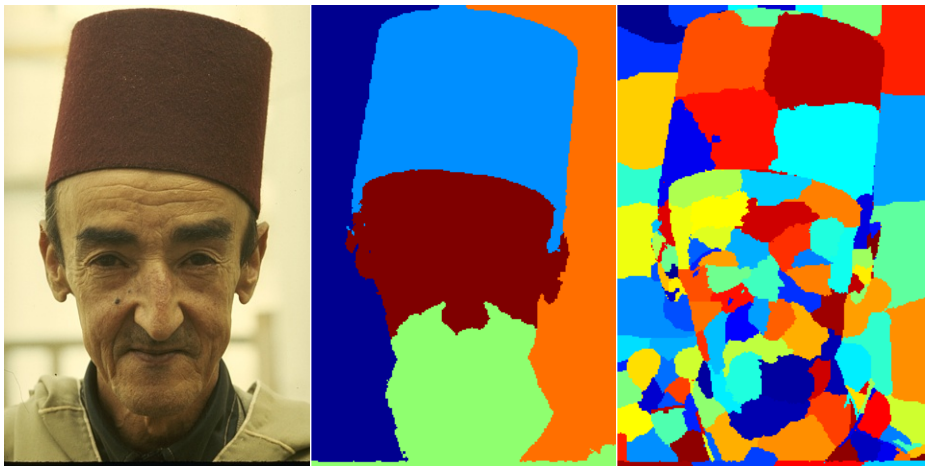


Figure 7-2: Left: Original Colour Image; middle to right: Segmentation results with N = 5 and 120, respectively

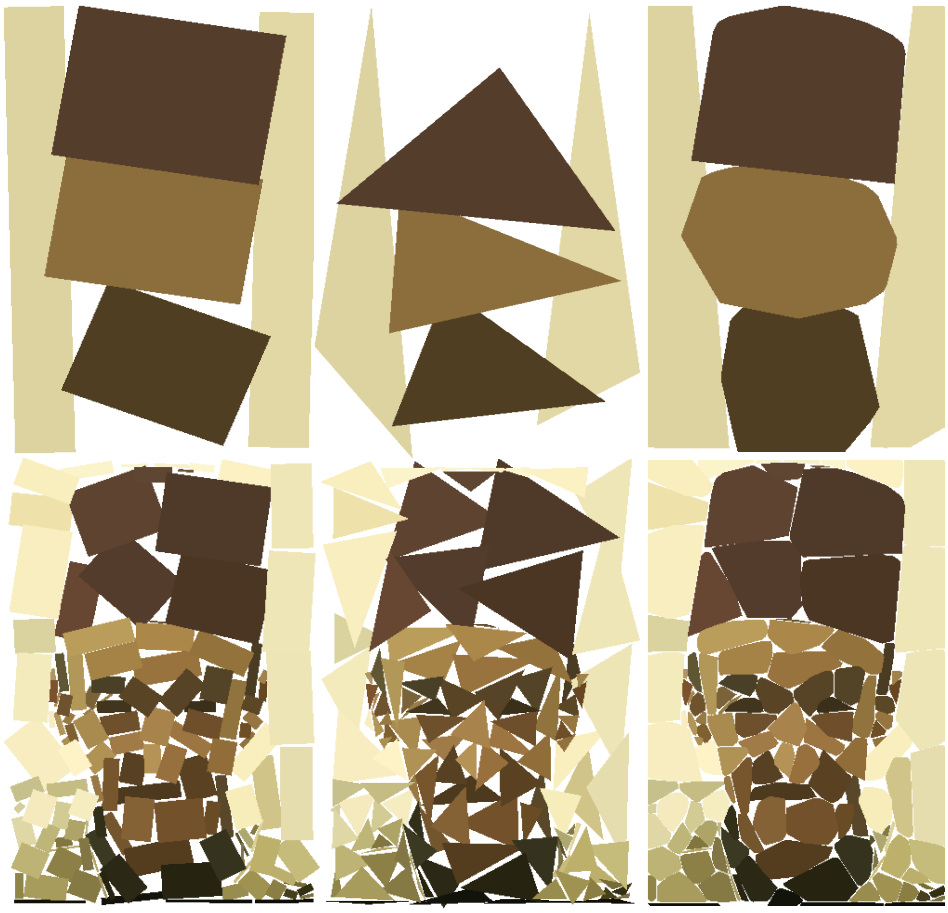


Figure 7-3: Results of fitting shapes of a single type. Left Column: Fitting rectangles; middle column: Fitting Triangles; right column: Fitting robust convex hulls

Figure 7-4: Results of automatic shape selection

of the convex hull to each segment in the image hierarchy. Figure 7-3 shows results of fitting some user-defined shapes to segments of both the foreground and background layers.

Having a set of shapes fitted to each image segment, the problem now is how to choose amongst them. As previously stated, this can be done through user interaction or automatically using the automatic shape selection technique proposed in Section 6.2.2. Figure 7-4 illustrates results of using the automatic shape selector on both layers of the image hierarchy.

7.2.3 Rendering Shapes

We can now fit shapes to each region and automatically select the best fit among those. Segmentation at a coarser level yields large and more abstract shapes; whereas, detailed segmentation often result in shapes that are too small and over detailed. What we really want is to preserve an appropriate amount of detail, while keeping the abstractness. We resolve this issue by treating the layer of larger/coarser shapes as "background" and the one with smaller and detailed shapes as "foreground". Doing this naively will result in the top layer completely covering the bottom one. We solve this by filtering the detailed shapes on the top layer by their corresponding shapes underneath. More specifically, we only

render shapes from the top layer whose colour deviates from that of the shape underneath above a certain threshold. Hertzmann also used colour differences to place strokes on top of those already painted in his stroke rendering work [61]. Unlike him, we measure colour differences in terms of just noticeable difference (jnd) in CIELAB colour space. For instance, take two colours, (L_1, a_1, b_1) and (L_2, a_2, b_2), we define their colour difference ΔE_{12} as follows

$$\Delta E_{12} = \frac{\sqrt{(L_1 - L_2)^2 + (a_1 - a_2)^2 + (b_1 - b_2)^2}}{jnd}$$

where $jnd \approx 2.3$ in CIELAB colour space [140]. Therefore, in general, ΔE measures how many jnds one colour deviates from another. By placing a threshold on ΔE, we can control the level of detail to render on the top layer; increasing the threshold results in fewer shapes being rendered and vice verse. The effect of increasing ΔE is illustrated in Figure 7-5.

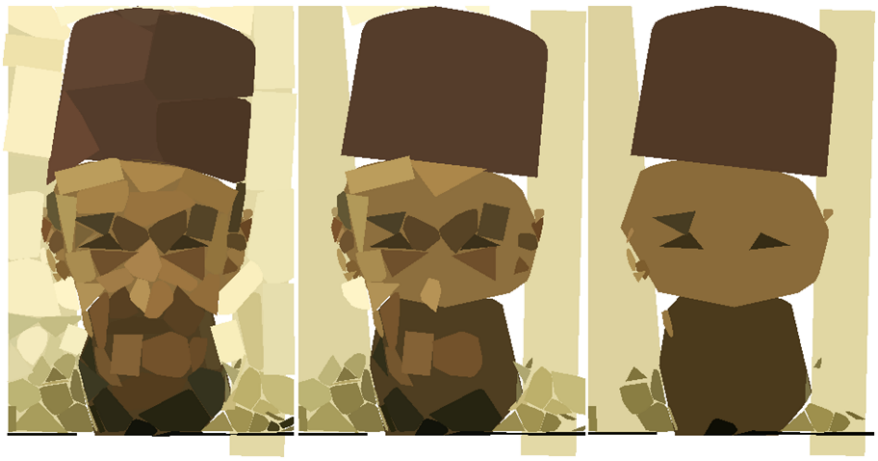


Figure 7-5: Result of combining two layers of shapes using different ΔE values. From left to right: $\Delta E = 0$, $\Delta E = 5$, $\Delta E = 10$.

The left of Figure 7-5 shows such a result of merging two layers of shapes without colour filtering on the foreground layer. It can be seen that on the left that the foreground layer completely covers the background one. The result in the middle however correspond better to a proper representation of a human face. It is clear that in the middle result, features like the "hat" reside in one single shape inherited from the bottom layer, whereas, details such as facial features are taken from the top layer. When ΔE increases, more foreground shapes get filtered out, leaving only a few behind. The effect of such is shown at the right of Figure 7-5. As with the result shown in Figure 7-5 and all other results in this section, a

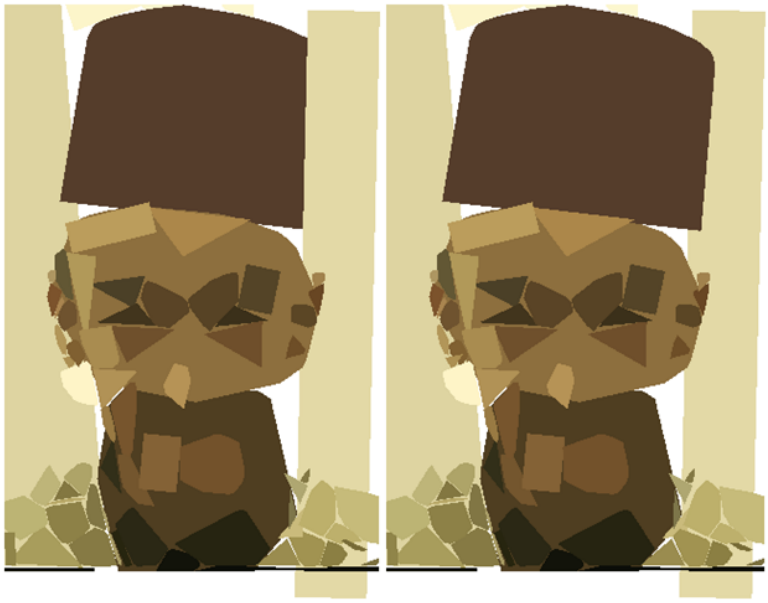


Figure 7-6: Left: Rendering shapes without the ordering introduced by shape fitting error τ; right: Now correct rendering result, after applying an ordering.

constant threshold of 5 is used on ΔE, which has been experimentally found to yield good results.

When it comes to rendering shapes into a framebuffer, the ordering of shapes can play a role in the final output as well. This is because of the fact that shapes fitted often overlap each other and which one comes on top can confuse viewer's perception. We tackled this problem by introducing a shape fitting error τ. Given a shape model s and its corresponding region r, we denote the area of s by S and similarly R for r, then τ is defined as the following ratio:

$$\bigcap(S,R)|/|\bigcup(S,R)$$

The idea is then to lay down shapes according to their assigned fitting error. Shapes with large fitting errors are rendered before those with smaller errors. The effect of how the proposed shape fitting error effect the order of rendering is demonstrated in Figure 7-6. It is clear that the "hat" is now correctly positioned above the background.

To create a paper cutouts effect of the shapes that appear in the later Matisse, exemplified by artworks such as "L'escargot", we simply counted the number of shapes lying over each pixel; the resulting height field then became a bump map.

Figure 7-7: Left: A rendering resembling a paper cutout effect; right: the same rendering as the right, but only transparent paper cutouts.

To create transparent paper we simply used "alpha" colour channel. In Figure 7-7, a paper-cuts rendering and its transparent version of the right of Figure 7-6 are provided.

7.2.4 Gallery of Renderings

This section exhibits a collection of shape rendering results. Divided into two parts, we first demonstrates examples of NPRP images that can be generated with no additional parameters other than a choice of rendering style such as "flat", "embossed" or "transparent". The second part includes a collection of synthetic art works that applies modern NPRP stroke-rendering techniques on top of our shape renderings. Specifically, we used the technique described in [147] to generate all results shown in this part.

A paper cutout rendering of "bird feeding" is shown in the middle of Figure 7-8, where the original image is shown on the left. As can been seen, there is a nice balance of shapes in the final rendering; relatively large entities in the scene such as the trunk of tree has a single rectangle fitted, while a combination of small shapes together compose the nest. A highly abstract version of "bird feeding" is shown on the right of the same figure, where the user chose to fit circles across

Figure 7-8: Left: Original colour image; middle: Shapes rendered as paper cutouts; right: An abstract result of fitting circles.

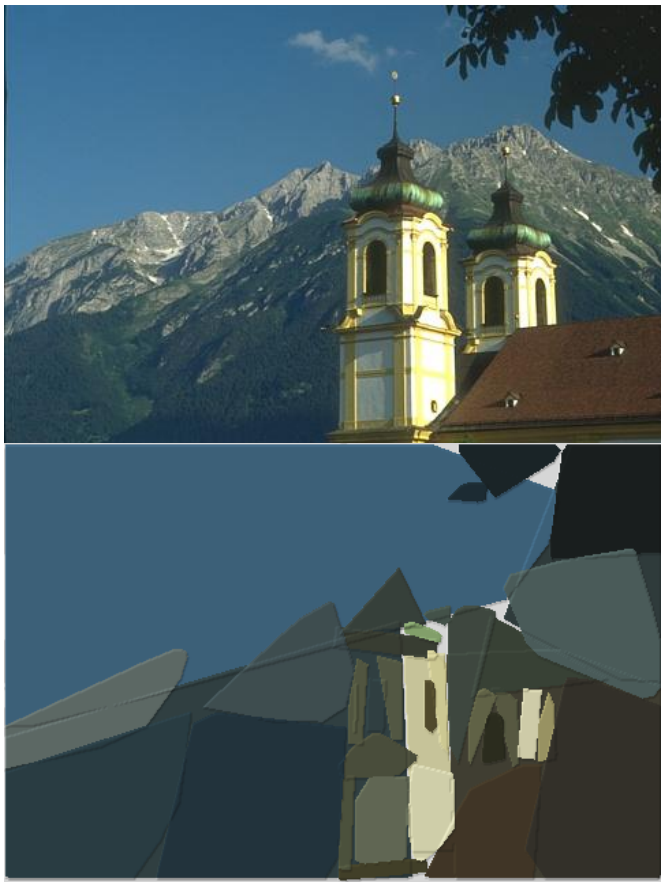


Figure 7-9: Top to Bottom: Original colour image and chapels rendered as paper cutouts.

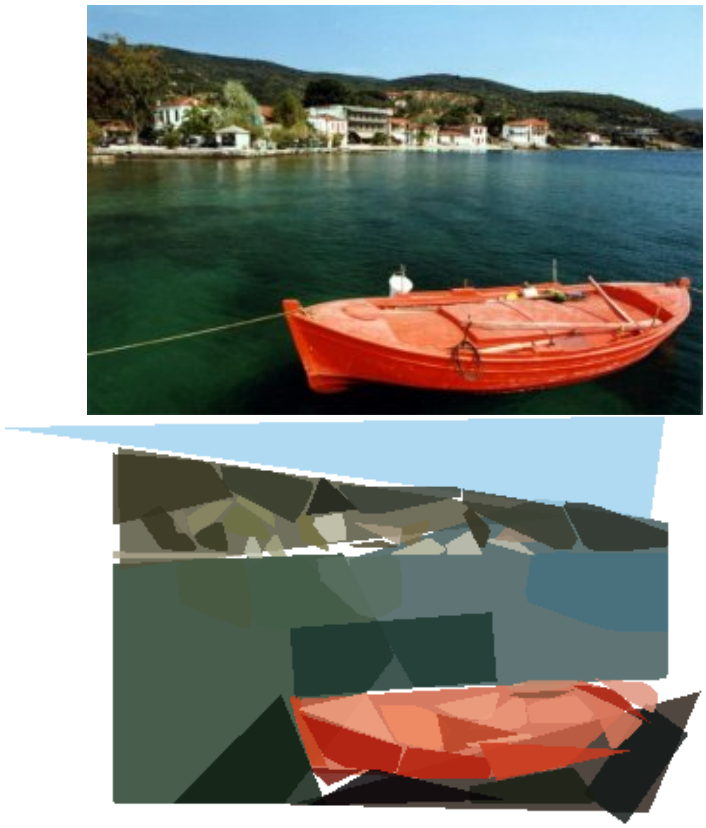


Figure 7-10: Top to Bottom: Original colour Image; shapes rendered as translucent paper cutouts.

Figure 7-11: Les Balloons

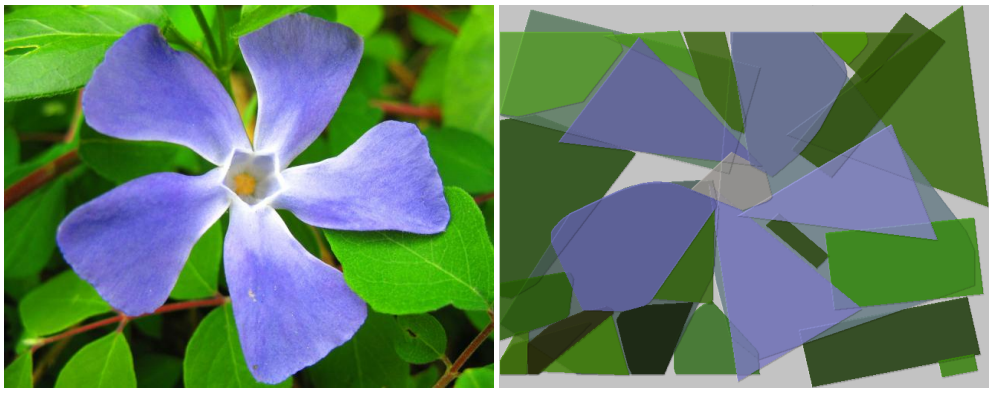


Figure 7-12: Flower rendered as transparent paper cutouts.

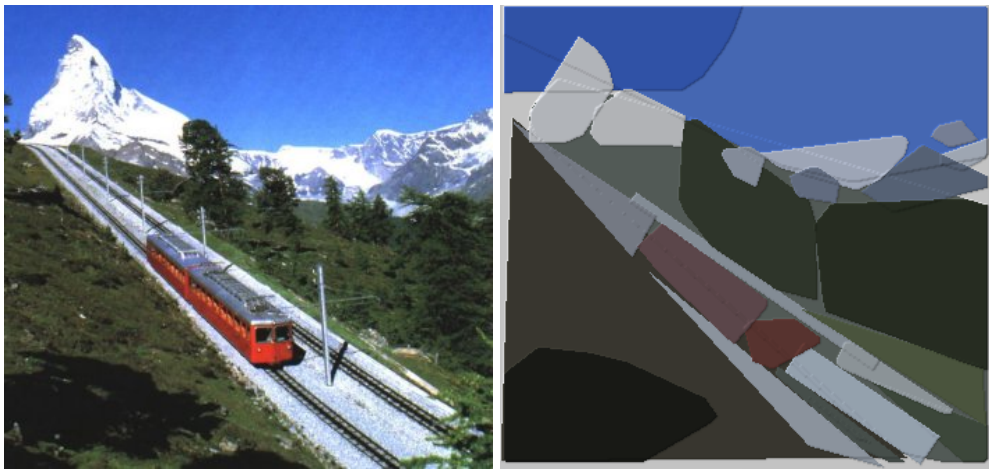


Figure 7-13: A landscape scene rendered as papercuts.

every region. This may not be to everyone's taste, but we liked it and the result is an extreme example of how shape simplification enables abstraction that goes well beyond stroke modelling.

In a similar fashion to the "bird feeding" example, Figure 7-9 shows how a landscape scene is rendered into a piece of artwork where paper cutouts were used as basic elements. Again, large objects such as the mountain at the back and the sky have rather large shapes fitted; but both "towers" are composed of a rather interesting combination of smaller shapes. The scene shown at the top of Figure 7-10 is rendered as a combination of transparent paper cutouts, shown beneath. It is interesting here to note how various shapes are fitted to represent the boat itself, whereas the "sky" is represented as a single triangle.

The method favours broad, clean colours; so examples such as hot air balloons (Figure 7-11), flowers (Figure 7-12) and red train (Figure 7-13), make pretty

Figure 7-14: Left to right: a painted man and a crayoned man

pictures.

All the above rendering results are obtained solely from shape simplification, with simple effects like paper cutouts put on top. But, of course, we can make of stroke renders to further enhance the aesthetic appeal of our synthetic artwork. Two oil paintings are included in Figure 7-15. Figure 7-14 shows both an oil painting and a crayon painting of the "man". The stroke rendering technique is adapted from Shugrina et al. [147], which operates on area segments of images. In our case, each fitted shape is treated as a segment of constant colour.

Finally, we offer up a rather whimsical version of the Matisse's snail, in Figure 7-16, as a reminder of the source of our motivation.

In summary, we have shown that by abstracting image regions into simple geometric shapes, we are able to move towards automatically creating more abstract art than was previously possible. More specifically, the art we synthesised was influenced by artists such as Kandinsky and later Matisse who advocate the use of geometric shapes. Shape simplification is the key to delivering the level of abstraction resembled in such type of art. Importantly, we can automatically select which shape fits the best among a few that we can fit. We are also able

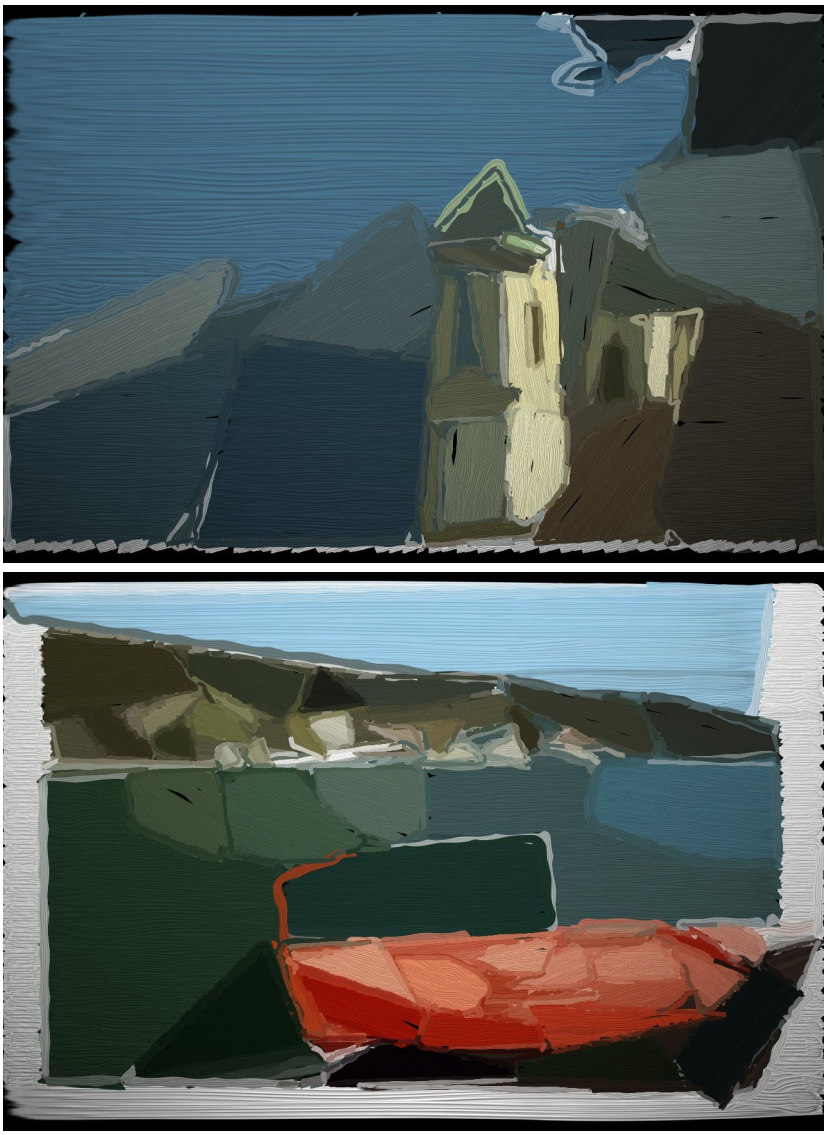


Figure 7-15: Two oil Paintings; above a church tower, below a red boat.

to combine shapes of various granularities and so preserve appropriate amount of detail. The whole process only takes two parameters during the segmentation step, these being integers to specify how many segments the user wants on each layer.

Nonetheless, the idea of using simple geometric shapes alone to represent image segments is limited. Artworks produced this way do not offer a global abstraction to objects for two reasons: image segments do not necessarily correspond to objects and their semantic parts; they do not necessarily carry any mutual relationships amongst them. Higher-level vision is required to understand the

Figure 7-16: The Snail

image in a more global and semantic fashion. In the next section, we will move on from rendering shapes alone and study how topological structures provide a step forward in the direction towards abstraction.

7.3 Towards Generating More Abstract Synthetic Art Using Object Structures

In the previous section, we investigated the use of shape simplification in terms of creating abstract synthetic art. We demonstrated that by fitting simple geometric shapes to image segments and rendering them into various styles, we are able to synthesis art of an abstract nature; styles of which advocated by artists such as Kandinsky and Matisse. In this section, we continue to study the problem of creating abstract synthetic art, with the aim to produce even more abstract renderings, in the styles of child-like drawings, Mirö and Picasso.

7.3.1 Moving onto Object Structures

We advocate the use of topological object structures in making synthetic abstract art for four main reasons:

- Master Artists such as Joan Mirö, Picasso often use topological object structures to abstract objects, evidence of which can be found in Chapter 1 of the thesis.
- In the previous chapter, we demonstrated the benefit of using object structures in terms of matching between photographs and paintings. The success in matching partially proves that object structures offer a high-level representation of objects, which is invariant across different depictive styles. We are in a good position to study how this high-level description can be used to synthesise art.
- The simple image hierarchy we previously used in the last chapter offers minimal abstraction to images. Although we have successfully demonstrated that abstract artworks can be generated by rendering simple geometric shapes alone, the idea of using them to represent image segments is still limited. This is because the abstractness they offer are local to their corresponding image segments. In addition, there is a lack of global relationship among the fitted shapes; each shape provides an abstract representation only to its corresponding segment. Although image segments themselves are already of a higher-level representation than pixels, the global understanding of the image/object is to be further explorered in order to make art of a more abstract nature. Such global relationships should be extracted using even higher level computer vision techniques.
- We have already developed a systematic way to extract object structures from images, including an image hierarchy editor to facilitate the object structure extraction process. Two hierarchical image descriptions were developed in turn, and we have shown that structures can be readily extracted from the latter.

In this section, we set out to prove our proposition that topological object structures offer an even more abstract perspective to computer-synthesised art. Figure 7-17 shows a flow-chart of the proposed abstract art generation process. We

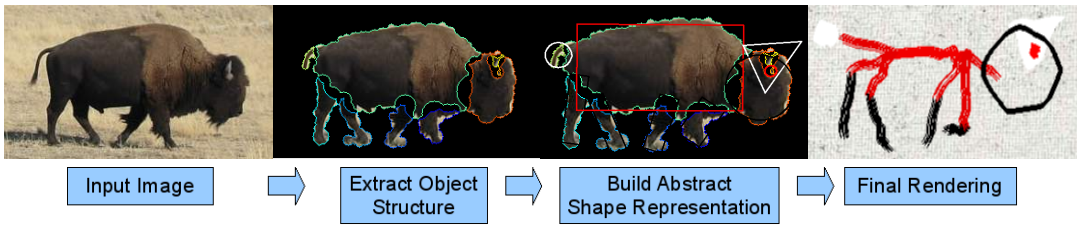


Figure 7-17: A typical abstract art generation process. The whole process took less than 1 minute, with 15 mouse clicks to extract the structure of the bison.

start by building a hierarchical image description, from which we can automatically extract object structures using the technique described in Chapter 4. The proposed structures can then be refined using the image hierarchy editor proposed in Chapter 5. Having a topological decomposition of the object, we continue to fit shapes to its parts. The choice over which shape best fits a given object part can either be decided by the proposed automatic shape selector (Section 6.2.2), or according to the user's personal preference. With each part described by a simple geometric shape, we can render this abstract representation of the object appropriately to synthesis abstract artworks. In summary, given an input image, there are three main steps to synthesis abstract renderings: (i) object structure extraction; (ii) building an abstract representation by fitting shapes to object parts; (iii) and using traditional NPRP techniques to render.

In essence, our approach is to process an input image into a model, and then render the model. Such model is an abstract representation of an image and is the key in capturing the abstractness with images. The model embodies two essential aspects of an object: (a) its *structure*, which is the way its parts connect together; and (b) the *canonical shape* of each part, meaning ellipse, rectangle, triangle *etc.* A graphical model of nodes and arcs is used to encode structure, while labels encode shape. This kind of model can be automatically extracted from the input image, and supports a wide range of renderings. Moreover, the model sits behind a powerful user interface that provides the user with support to adapt and choose the look-and-feel of the art they synthesise.

7.3.2 Extracting Topological Object Structures

This section summarises the process of extracting object structures from an input image, which is the first step towards building abstract models of images. Detailed

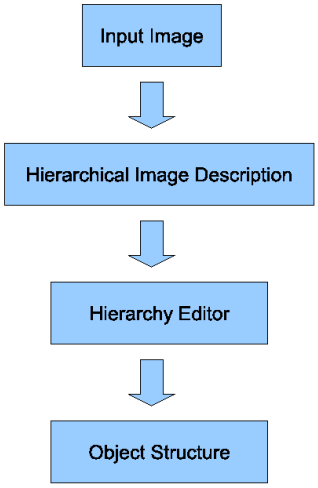


Figure 7-18: A typical object structure extraction process

descriptions of the structure extraction and editing processes can be found in Chapter 4 and 5 of this thesis, respectively. A step-wise illustration of the entire object structure extraction process in provided in Figure 7-18.

Based on the technique described in Chapter 4, we can automatically extract object structures, by first building hierarchical image descriptions then parsing them using graph energy. Afterwards, using the editor detailed in Chapter 5, the desired object structures can often be obtained using a few mouse clicks. Using the editor, a user is able to select the object he/she wants from the image, define its parts and establish relationships amongst them. Another important property of the editor lies with the flexibility that it introduces to the structure extraction process. Such flexibility also facilitates one's desire to be creative. Figure 7-19 demonstrates two possible interpretations of the "face", and how the proposed structure extraction framework is able to accommodate such ambiguity. The left column of Figure 7-19 shows a situation where the user decided to give a coarser interpretation of the "face", which consists of one facial area, two eyes, two ears and one mouth; whereas, another user obtained a more detailed decomposition of the face, by further breaking the facial area into a forehead, an eye area, a nose and a chain. The corresponding graph representations of both object structures are shown below each face.

We can also enforce a link type to the relationship between two object parts. There are normally two types of link types: "abutting" and "containing". Given

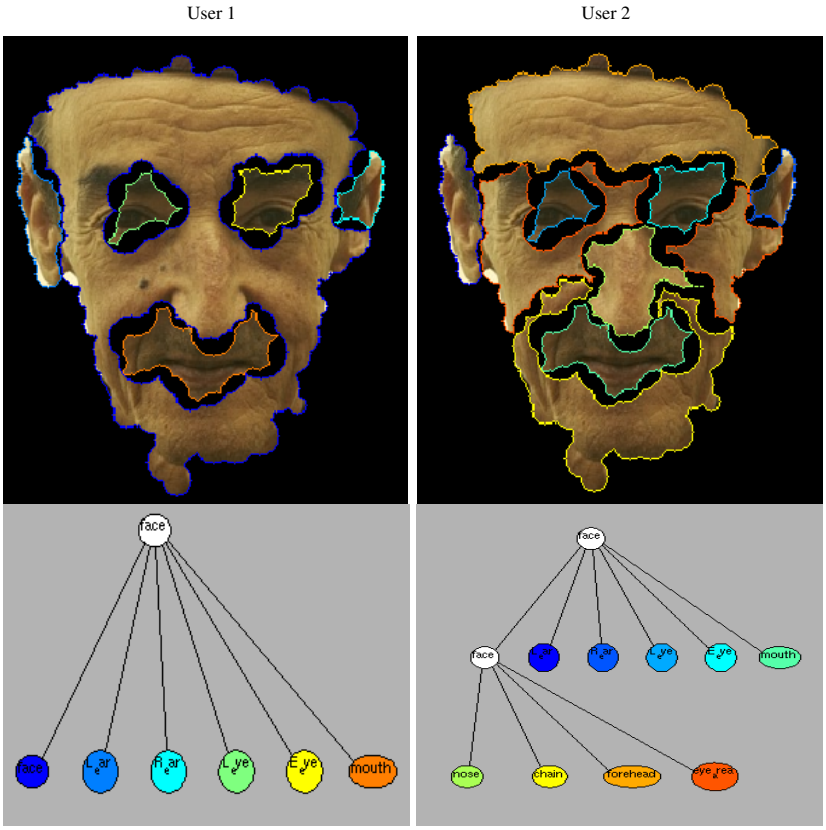


Figure 7-19: Different topological object structures extracted by two users using the hierarchy editor tool.

two parts of an object A and B, we define:

A "contains" B: When B's outer boundary is shared by A's inner boundary. In other words, B's area is spatially surrounded by A. If A "contains" B, then B "contains" A is false, in this case, we can say B is "contained" by A.

A "abuts" B: When A and B share a common boundary and A does not "contain" B and vice verse. If A "abuts" B, B also "abuts" A.

Figure 7-20 provides examples of different link types between object parts, where the links are overlaid on top. "Abutting" links are shown as undirected links; whereas, "containing" arcs are directed because of their one-way nature. As can be seen, just like all of our faces, ears are abutted to the facial area, which contains two eyes and one mouth.

Having object structures in hand, in the next section, we will explain how we can

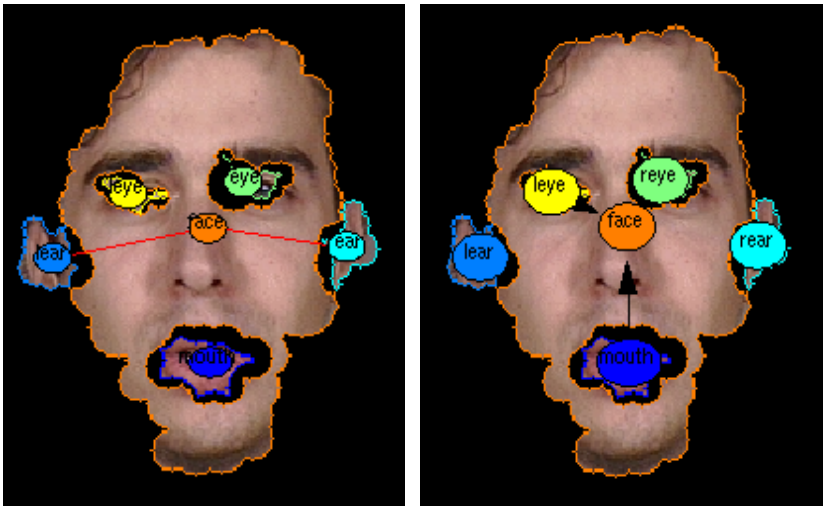


Figure 7-20: Two types of links between object parts. Left: Abutting links; right: Containing links.

combine the abstractness that object structures offer with that offered by shape simplification, so to complete building the proposed abstract models of objects.

7.3.3 Building Abstract Models of Objects

Previously in Section 6.2.1, we demonstrated how simple geometric shapes, such as triangles, rectangles and so on, can be fitted to image segments. An automatic shape selector was proposed at the same time, which can be used to choose the "best" shape among a few (Section 6.2.2). Nevertheless, back then image segments were obtained from conventional image segmentation techniques such as Normalised Cut, which offer minimal abstraction to the image/object; image segments normally conform to some sort of homogeneous property, such as colour similarity, etc. Moreover, a simple image hierarchy had to be used in order to preserve appropriate amount of details. Artworks produced this way do not offer a global abstraction to objects, because such image segments do not necessarily correspond to semantic objects and their parts, and there is a lack of spatial relationship among them.

We proposed to use object structures to tackle the above limitations. Object structures extracted are simply image segments with mutual relationships in-between. In contrast with the simple image hierarchy used previously to synthesis "shapy" art (Section 7.2), image segments that make object structures do

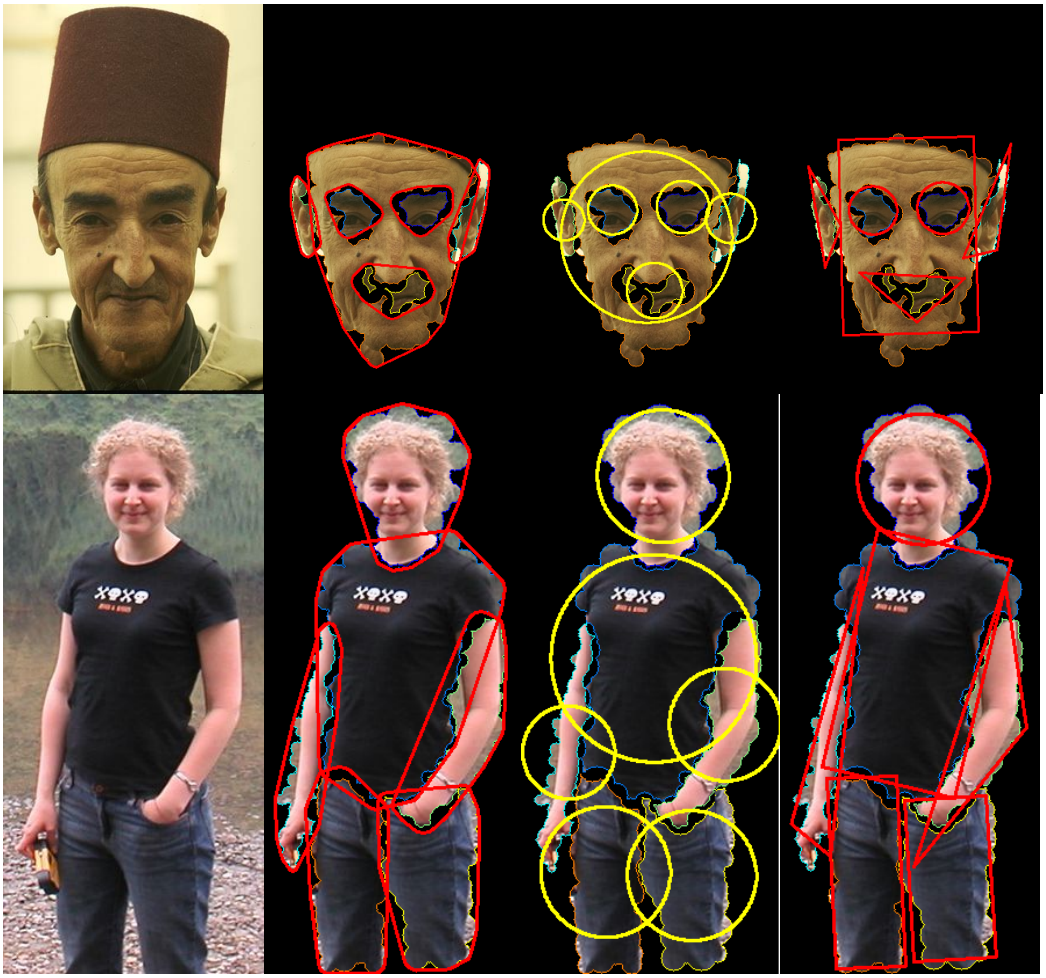


Figure 7-21: From left to right: Original colour images; fitting robust convex hulls to parts of the objects extracted from their corresponding image descriptions; fitting circles; fitting user-defined shapes to parts.

correspond to semantic parts of objects and each pair has a link assigned. Because object parts are just image segments, we can use the same shape fitter and selector detailed in Section 6.2 to assign shapes. After shape fitting, our abstract object models are built, where object parts are abstracted as simple geometric shapes and these shapes inherit the same topological relationships from the original object structure. This model can be encoded as a graphical model of nodes and arcs: nodes are labelled with shape labels and arcs correspond to part relationships. Figure 7-21 shows two examples of fitting shapes on top of previously extracted object structures, where shape labels were manually selected.

Having abstract object models in hand, we will demonstrate how such models can be rendered to create abstract artworks of an even more abstract nature.

Figure 7-22: Two examples of Mirö paintings. Left: "Birds, 1973"; right: "Day Break, 1968".

7.3.4 Rendering of Abstract Object Models

Within an abstract object model, at a global level, objects are abstracted as a collection of parts; whereas, at a local level, object parts are abstracted as simple geometric shapes. We are able to synthesise abstract artworks using these models; and because of the abstractness captured within them, the resulting synthesised artworks will be more abstract than artworks in the previous section. In this section, we describe how abstract models of objects can be rendered to create abstract synthetic artworks, particularly the type of abstract art that master artists such as Mirö and Picasso advocates, or those in the styles of child-like drawing and cave paintings.

In order to mimic the artistic styles of Mirö or Picasso, or indeed child-like drawings, we need to look closely at their artworks, so to identify the main characteristics captured within. Here, we will concentrate on analysing Mirö's paintings as representatives. Mirö combines simple entities, such as general shapes and lines, to create objects. It is the relationship among such elementary entities that is important to what is perceived in his paintings. A bird would normally have a body and two wings and a woman would have a head, a body, arms, legs and so on. If we have a closer look at "Birds, 1973", shown on the left of Figure 7-22,

most of the birds Mirö drew have two "wings". In contrast, the human figure Mirö painted in "Day Break, 1968", shown on the right of Figure 7-22, has a clear head, body, legs, separation. In Mirö's paintings, it is the structure of objects that differentiate them. We already have means of extracting object structures from images, it is how Mirö depicts object parts that interests us. In summary, the following characteristics can be readily found in Mirö's artworks:

- Shapes filled with plain colours
- Shapes drawn as closed boundary curves
- A heavy use of lines and curves
- Strokes and shapes are often laid out in a structured way, in accordance with topological object structures
- Only the most salient parts of an object are drawn
- Background rather plain then textured

We argue that once the above core characteristics are followed when synthesising abstract artworks, Mirö-style renderings can be achieved. Please note here that we are not aiming to re-produce artworks of either Mirö or Picasso, but create novel abstract artworks from images that capture their ways of abstracting.

After observing the main characteristics in Mirö's paintings, we shall come back to analyse the abstract models we obtained from objects depicted in images, so that specific ways of rendering of such models can be decided accordingly. Our abstract models of objects already carry the two most important elements in Mirö's abstract artworks, i.e. topological object structures and simple geometric shapes. Therefore, the only step left is to render these abstract models into various artistic styles.

In accordance with the main characteristics found within Mirö paintings, we offer three different ways to render each shape in a given abstract object model:

1. Shapes as distorted contours of uniform colours
2. Shapes as filled distorted contours of uniform colours
3. Shapes as long and curly strokes

Figure 7-23: A pure geometric shape and its three rendering styles

Figure 7-23 illustrates the three different styles a shape can be rendered into. Each shape in the abstract model can be rendered separately and later composited according to topological object structures. We will now describe how shapes can be rendered into the above three different styles.

The first two rendering styles, rendering shapes into contours and filled contours of a single colour, are relatively straight forward. The only aspect that needs more attention here is the shape boundary distortions that are seen in Mirö's paintings. Such distortions are often sourced from two possible causes: hand movement and the artist's intention towards abstraction. In real paintings and drawings, it is natural for lines/strokes to appear distorted, in Mirö's paintings such distortion appears stronger, accommodating the artist's intention towards abstraction. Accordingly, there are two challenges in creating the sort of distorted shape boundaries: mimicking distortions that is introduced by human hand movement and that put in by the the artist to exemplify abstraction. We propose an integrated way of mimicking hand movement and introducing extra distortion to shape boundaries based on curve fitting.

Creating Stroke and Contour Distortions

With circular shapes such as circles, ellipses and superellipses as exceptions, each side of a polygon shape is a line segment $L(a, b)$, where $a = (x_0, y_0)$ and $b = (x_f, y_f)$ are the starting point and end point, respectively. For each such line segment, we first generate its parametrised trajectory using the Flash and Hogan [45] model as follows:

$$x(t) = x_0 + (x_0 - x_f)(15t^4 - 6t^5 - 10t^3)$$

$$y(t) = y_0 + (y_0 - y_f)(15t^4 - 6t^5 - 10t^3)$$

where the value of t varies from 0 to t_{final} and the number of points on the final trajectory is decided by a time-step parameter δt. The larger δt is, the fewer points will be sampled along the line and the choice of δt is affected by the length of the line. The other parameter, t_{final}, is also related to the number of samples on the trajectory, but can be set as a constant as δt can always be adjusted in proportion. In this thesis, we apply a constant value of 2 to t_{final} upon using the Flash and Hoganin model. Flash and Hogan [45] first recognised the relationship between the choice of δt and the length of the line and have shown by experiment that short hand drawn lines tend to be perceptually closer to straight lines, hence can be modeled by coarser samples along the trajectory. This argument is also supported by Meraj et al. [95], who experimentally verified the change in δt relative to line length. The line trajectories created by Flash and Hogan model offer "the smoothest motion to bring the hand from an initial position to the final position in a given time" [45] and has been successfully applied in creating realistic pencil lines [95].

Although the Flash and Hogan trajectories offers reasonable realistic human hand-drawn trajectories, the distortion it introduces to the shape boundary is limited, in comparison with what Mirö applies to his shapes. In order to accommodate more distortion into the model, we offer the option to add more noise to the Flash and Hogan trajectories as a second-phase of our shape distortion method. We do this by selecting a random subset of samples along the Flash and Hogan trajectories, based on a user-defined threshold $\gamma \in (0, 1)$. The larger γ is, the more samples are chosen; so when $\gamma = 1$, the original trajectory is preserved. For the N points randomly selected, we add some random Gaussian noise to tabulate the positions of those N points, yielding another set of new points N'. An interpolating spline fitted to N' then becomes our new distorted representation of the input shape. We always set the mean of the Gaussian noise to be zero, therefore, given a δt for the Flash and Hoganin model, only two parameters affect the actual distortion of the shape: γ and σ; γ is a threshold on sampling rate and σ is the variance of Gaussian noise. Between those two parameters, increasing γ means a more jagged boundary, resulting from a denser sampling; whereas, more intensive Gaussian noise always lead to more distorted paths. Figure 7-24 demonstrates the effects of changing γ and σ.

As previously mentioned, the proposed noise model can be applied to polygon shapes, which consist of a finite set of line segments. For circular shapes, we can

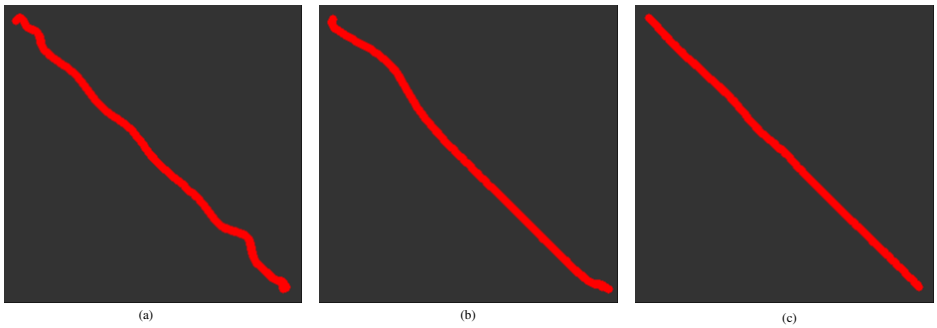


Figure 7-24: Effects of changing γ and σ in the distortion model. (a): $\gamma = 1$, $\sigma = 3$; (b) $\gamma = 0.3$, $\sigma = 3$; (c) $\gamma = 0.3$, $\sigma = 1$.

simply parametrise them without the Flash and Hogan line model and apply the rest of the distortion method in the same way.

Rendering Shapes as Curly Strokes

We can now render pure geometric shapes into distorted contours, empty or filled. This leaves us with only one particular rendering style to consider, that is, rendering shapes as a single descriptive curly stroke. As previously mentioned, Mirö tends to use lots of such strokes to represent object parts and sometimes even the whole object. They are used because Mirö treats them as abstract representations of objects. In our own abstract object representation, i.e., shape-based representation, we abstract objects and their parts in terms of pure geometric shapes. The question then comes to how can we abstract a shape, so that it can be described as a line or a curve? The answer here is medial axis transform [85]. The medial axis of a 2D shape is defined as the locus of the centre of all the maximal inscribed circles of the shape. Medial axis are often used as a complimentary representation of shapes but in more abstract and simpler forms. Medial axis transform leads to an abstract shape representation widely refereed as shape skeletons. Shape skeletons provides unique descriptions of shapes and the original shapes can be re-constructed given their skeletons. Figure 7-25 demonstrates skeletons fitted to every shape within an abstract model. All we have to do now is to select an appropriate subset of the medial axis of a shape and use that as the path of the stroke describing the shape, and by transition the underlying object. We currently use the longest continuous section on the medial axis to describe a shape. In the special case of a circle, we simply use its diameter of a random orientation to describe it. Once a shape is mapped into a line/curve, we can

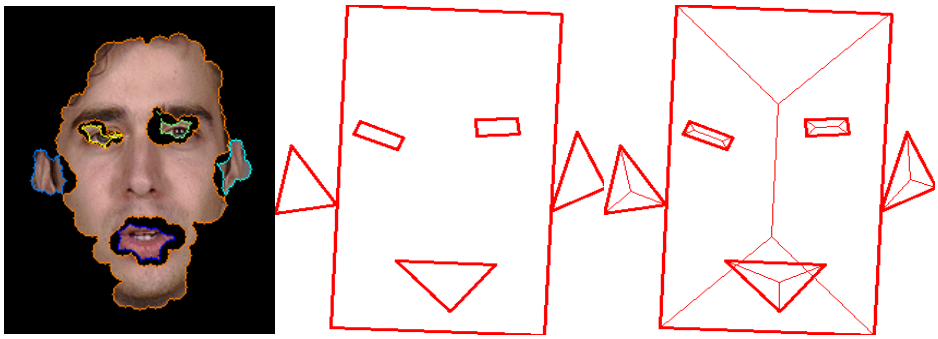


Figure 7-25: Left to right: Structure of a face, its abstract model and shape skeletons

employ the previously proposed distortion model to generate its final trajectory.

A GUI to Facilitate the Art Synthesising Process

We can now render shapes into the three main characteristic styles found in Mirö's paintings, viz, distorted shape countours, distorted filled shapes and single strokes. Overall, given an image, we can: (i) obtain its structure; (ii) build the corresponding abstract model; (iii) render it in accordance with the three main characteristics. Being able to follow these steps to get a final rendering marks the end of the description of the proposed abstract art generation process. All three steps have default options so that the user can potentially turn an image into a crude abstract rendering of it by clicking on a "go" button. If the user wants to be more specific and/or creative, he/she would need to alter the abstract models and define rendering styles for each shape; a graphical user interface (GUI) is implemented to facilitate this process.

Our user interface is important in allowing users to correct any modeling errors, exchange canonical shapes, and select rendering options. The interface has access to the complete abstract model. The user can edit the automatic shape by choosing from a list which is wider than just the canonical shapes. The list offers special restrictions on canonical families, so ellipses (canonical) and circles (restriction) both appear. Once the user is happy with the model (structure and shape) of an object, it can be rendered by clicking on the "render" button.

The interface allows choice of many rendering options as well, the effects of some of which are demonstrated in the Gallery Section 7.3.5. We can render both arcs and nodes of the model individually. Rendering arcs is to render the structure of

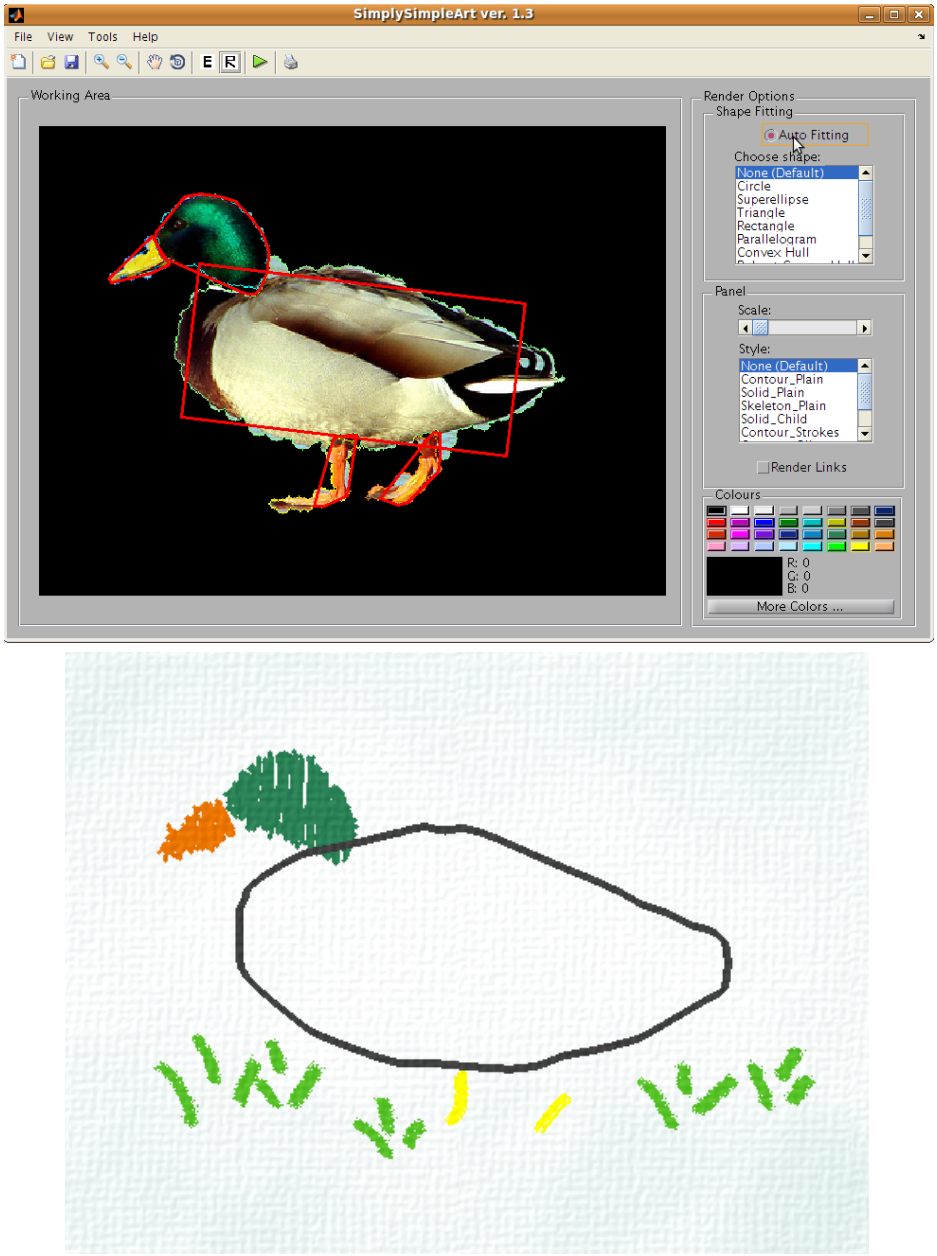


Figure 7-26: Top: a screenshot of the GUI that facilitates the art synthesising process; bottom: an abstract rendering produced using the GUI.

the object, whereas rendering nodes is to render its shape. An array of standard media emulations is built-in to the interface: pencil, paint, chalk and others.

A screenshot of the GUI is provided at the top of Figure 7-26; using the GUI one can easily create novel abstract artworks from images, such as the one show at the bottom of the same figure. In order to fine-tune the quality of the final rendering, so as to maximise its aesthetics, the interface also provides basic functionality

such as specifying a background, choosing colours, scaling shapes and so on. The user is able to interactively review the final rendering using a single click; and go back and forth to make changes to the rendering styles.

In the following section, we present several abstract renderings that several people created using the proposed GUI.

7.3.5 Gallery of Renderings

The abstract models of images support a wide range of simple art, as the examples shown here make clear. Each of these images was made by processing a photograph showing a single object. The automatic system was able to produce a good first model on most occasions, so that the user had to employ just a few minutes effort to edit out errors — such as limbs being lost to the background (Figure 7-27). If small details were required, such as eyes (Figure 7-27), then the effort rose, which extends editing time to about 3 minutes.

Once in place the user is free to select amongst a wide variety of options to

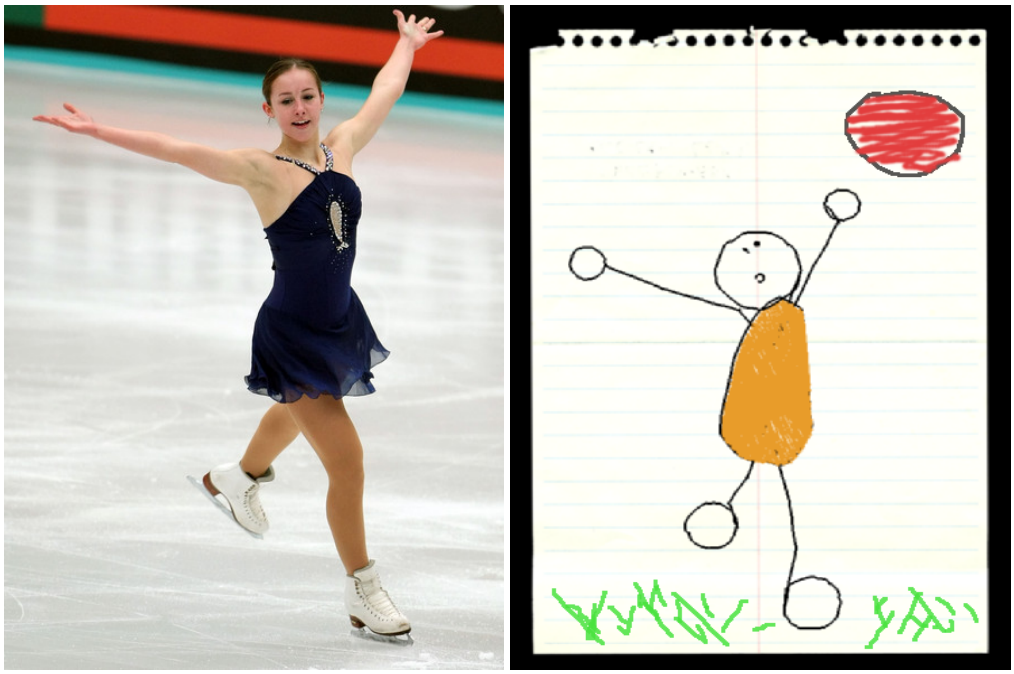


Figure 7-27: A skater rendered into a child's drawing on note paper. 33 mouse clicks were used to obtain the abstract model, most of which spent on extracting facial features. A child-scribble filling style was implemented and used to fill shapes.

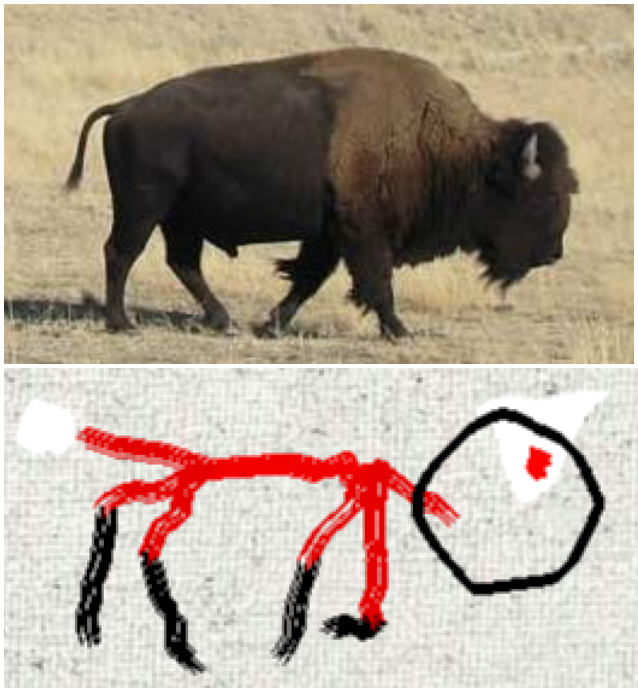


Figure 7-28: A bison rendered in the style of Mirö. The abstract model took 15 mouse click to complete, mostly spend on extracting the horn.

Figure 7-29: A human figure rendered as stickman. This model took 15 mouse clicks to edit into suitable form. It has been rendered as ink on paper. The whole process took about two minutes.

Figure 7-30: A horse rendered as rock art. Abstract model of the horse was built using 7 mouse clicks and standard bump mapping techniques was used to render.

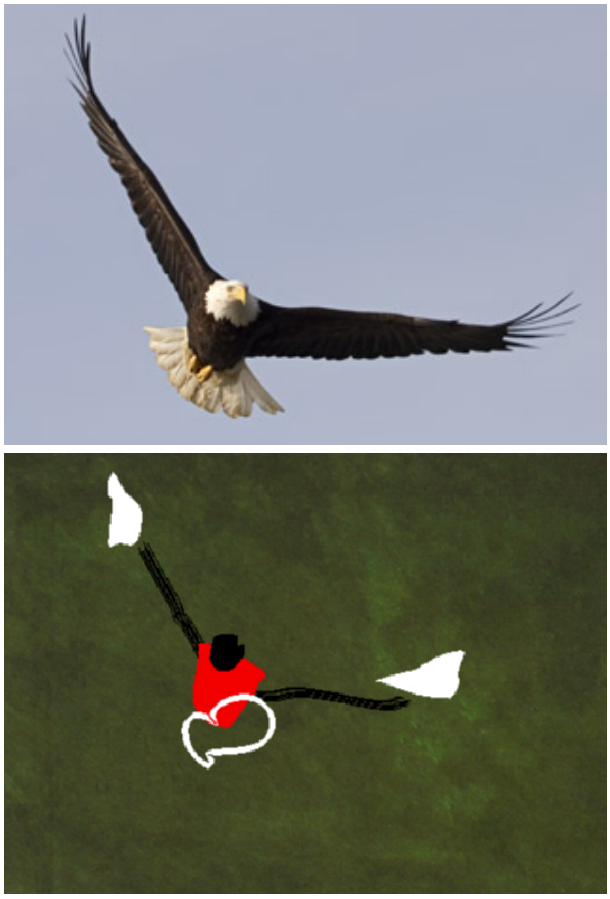


Figure 7-31: An eagle has been rendered in a style emulated after Mirö. The model took 7 mouse clicks to complete from the automatic model. It was rendered on canvas using oil media. The image took about one minute to make.

Figure 7-32: This is composition in the style of Mirö. It was made by composting renderings of each of the input photographs. The models required an average of about 15 edits each. The whole image took about seven minutes to make.

render areas as solid or boundary, to select colour, and drawing media. Specialist rendering styles were written to carve objects into rock. Each example given here shows the source photograph, states how long model editing took, and briefly explains the rendering options used.

The large pictures (Figure 7-32) with more than one object in them were made by compositing individually made paintings.

7.4 Conclusions

In this section, we investigated how artworks of an abstract nature can be synthesised using none or minimal user interaction. In particular we first tested the proposition that shape simplification is able to deliver a degree of abstraction that was previously impossible in the NPRP literature. In addition, we demonstrated how abstract artworks of the styles advocated by master artists such as Kandinsky, Melvich and Matisse can be synthesised. We then extended the system by advocating the use of object structures and show that by combining topological object structures with shape simplification, art of an even greater abstract nature can be created. Furthermore, we have developed an integrated system with intuitive GUI to facilitate the art creation process.

We conclude that both shape simplification and toplogical object structures provide a degree of abstraction in terms of synthesising abstract artworks. In particular, shape simplification provides local abstraction to object parts; whereas, toplogical object structures deliver a global and high-level abstraction to objects as a whole.

Part IV

Conclusions

Chapter 8

Conclusions

In this chapter we will first summarise the contributions of the thesis, then continue to make some conclusions based on the observations made from using the proposed algorithms, and discuss potential avenues for future research.

8.1 Summary of Contributions

We have proposed two hierarchical image descriptions and shown topological object structures can be readily extracted from the latter. Structure extraction is either done automatically using a novel graph theoretic measure or using a manual image hierarchy editor.

Afterwards, we have studied the use of object structures in two novel applications: clustering objects of different depictive styles and synthesising abstract synthetic artworks from photographs. Both applications demonstrate the benefits of object structures from their own perspective. Success in clustering objects regardless of depiction shows that object structures are invariant across depictive styles; whereas, success in synthesising abstract artworks from photographs demonstrates that object structures capture an appropriate degree of abstraction that is perceivable by humans.

In the rest of this section, we summarise the major contributions of this thesis:

- We have developed a hierarchical image description based on perceptual

grouping. We show that by searching for simple and stable groupings of image primitives, salient objects can be readily found from images.

- We have proposed a novel experimental setup for qualitatively evaluating the quality of primitive groupings. It also has potential applications in evaluating image segmentations, which are eventually groupings of pixels. The novelty of the experimental setup lays with the fact that human disagreement is used as a unit measure.

- We have developed a second image description based on hierarchical agglomerative clustering. It builds on the work of Haris et al. [59], but differs in two significant aspects: merging primitives are modelled as distribution of feature vectors and a novel graph theoretic stopping criteria is proposed to halt the merging process.

- We have demonstrated that using the same graph theoretic stopping criteria, we are able to automatically break objects into their parts. Both objects and their parts correspond to branches of their corresponding image descriptions.

- We have implemented an image hierarchy editor for users to interact with the automatically generated object structures. The benefits of the editor are two-fold: firstly, errors from the automatic output can be accommodated; secondly, different interpretations of object structures are made possible.

- We have shown that by encoding object structures into feature vectors of fixed lengths and clustering them in feature space, we are able to classify objects regardless of their depictive styles.

- We have shown that by combining object structures with simple geometric shapes, we can synthesis child-like drawings and abstract artworks in the styles of Joan Mirö, Wassily Kandinsky and Henri Matisse. An automatic shape selector is developed to choose the "best" fit shape among a few.

- We have implemented an intuitive GUI for the art synthesis application, using which users can break objects into their parts, fit shapes to them and select appropriate rendering styles.

8.2 Conclusions and Future Work

In this thesis, we set out to study the profound relationships amongst objects in different depictive styles, i.e., photographs, drawings and paintings. We proposed that object structures can be used as an invariant property that is key in linking objects of different depictive styles. In order to prove our proposition, we studied how object structures can be extracted from images of different kinds and consequently demonstrated their values in two novel applications: classifying objects across depictive styles and synthesising abstract art from photographs. The successes found in both applications provide partial evidences to the core argument this thesis makes, which is, topological object structure is an essential abstract property of objects.

We have shown that it is possible to extract object structures in an automatic fashion, by first building an image hierarchy and decomposing it using a novel graph theoretic measure. On the subject of forming clusters, spectral graph theory has been proven to be useful, using which graph representations of object structures can be encoded into feature vectors of fixed lengths. These clusters are found to make equivalence classes wide enough to cross depictive boundaries; yet sufficiently discriminative to be meaningful. In particular, it is interesting to observe that some of these clusters map to object classes of a broader semantic category. For example, our classifier recognises both "cows" and "horses" as "four legged animals"; a result in support to our main argument over the use of structures, because both animals share the same structures.

On the subject of extracting object structures, we believe potential improvements can be introduced to both of the proposed hierarchical image descriptions. The perceptual grouping based image hierarchy (Chapter 3) deserves some attention in the future, especially along the lines of studying the mutual relationships amongst Gestalt laws. In Chapter 4, we demonstrated that it is possible to decorrelate feature vectors using ICA and show that grouping performance can be improved this way. Although ICA only assumes linear correlations, we feel that the lesson here is that supervised learning is able to capture the hidden relationships between Gestalt laws. In addition, our second agglomerative clustering based hierarchical image description can benefit from: (i) An alternative set of primitives that is invariant to affine transformations of images; (ii) Using other information such as texture and shape as part of the feature vector describing an image primitives

and better modelling of distributions of such vectors; (iii) In particular, attention should be paid to the graph theoretic stopping criteria that we used to halt the grouping process. This particular technique is not restricted to be used in our particular case, but applicable to all grouping algorithms assuming graph theory.

There are several directions of future work that can be taken along the lines of classifying objects across depictive styles. First, it would be interesting to see how the current classifier generalises to more object classes and whether the integrity of wide semantic classes such as "four legged animals" will be kept. For instance, would a "tiger" fall into the class of "four legged animals" or a class of its own. More rigorous testing of such classifiers would involve building a testing benchmark system of our own, such as the Caltech-256 [52] database that is widely used for photograph-based object categorisation systems. Our database should contain a healthy mixture of photographs, paintings and drawings of objects, other than mainly photographs. Second, the question of how to break these semantic classes further also deserves some attention. For example, given a class of "four legged animals", how can other properties, such as colour and texture, be incorporated to distinguish "horses" from "cows". We have partially addressed this issue by augmenting object parts with shape labels, however, there is yet not enough evidence to make any decent claim on this. Third, the use of spectral graph theory to encode structures could be better addressed. A simple Laplacian matrix based description was used in our classifier, which was proven to work reasonably well on clustering objects from 13 different object categories. There are alternatives ways [144] of encode graphs in the spectral graph theory literature. It would be interesting to see how the current way of encoding scales as the number of object classes increase; and how might other alternatives cope with such change.

From a NPR point of view, we have shown that the abstractness carried by topological object structures is essential in synthesising abstract artworks from photographs. We have demonstrated how artworks in the styles of child-like drawings, Mirö and Picasso, can be rendered. It would be interesting to explore how yet more abstract artistic styles can be achieved, for example, by introducing a wider variety of brush styles and painting media. In addition, user interaction is currently needed to specify rendering styles for each part of the object, such as its shape and colour, so to make visually pleasing art. Possible ways to automate this process can be considered here as future work. Apart from object structures,

we have demonstrated that shape simplification is useful in delivering abstraction to computer synthesised art. However, we have not really touched on the topic of combinations of shapes, that is, how should shapes be combined and arranged in a way that is aesthetically pleasing. In addition, it is also worthwhile to extend the capability of our current NPR system to abstract videos.

Finally, we conclude that the structure of objects is indeed an invariant and abstract property that is shared by objects of different depictive styles.

Bibliography

[1] Akaike, H. (1974, Dec). A new look at the statistical model identification. *IEEE Transactions on Automatic Control 19*(6), 716–723.

[2] Bai, X., Y.-Z. Song, and P. M. Hall (2007). Learning object classes from structure. In *British Machine Vision Conference 2007*, Warwick, UK, pp. 322–330.

[3] Bangham, J. A., S. E. Gibson, and R. Harvey (2003, September). The art of scale-space. In *Proc.* 14^{th} *British Machine Vision Conference (BMVC)*, Volume 1, pp. 569–578.

[4] Baumberg, A. (2000, 13–15 June). Reliable feature matching across widely separated views. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Volume 1, pp. 774–781.

[5] Belhumeur, P. N., J. P. Hespanha, and D. J. Kriegman (1997, July). Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*(7), 711–720.

[6] Biederman, I. (1987, Apr). Recognition-by-components: a theory of human image understanding. *Psychol Rev 94*(2), 115–147.

[7] Bousseau, A., M. Kaplan, J. Thollot, and F. X. Sillion (2006). Interactive watercolor rendering with temporal coherence and abstraction. In *NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 141–149. ACM.

[8] Brooks, S. (2006, Nov.–Dec.). Image-based stained glass. *IEEE Transactions on Visualization and Computer Graphics 12*(6), 1547–1558.

[9] Brooks, S. (2007). Mixed media painting and portraiture. *IEEE Trans. on Visualization and Computer Graphics 13*(5), 1041–1054.

[10] Brosz, J., F. F. Samavati, M. T. C. Sheelagh, and M. C. Sousa (2007). Single camera flexible projection. In *NPAR '07: Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 33–42. ACM.

[11] Buchanan, J. W. (1996). Special effects with half-toning. In *Proc. Computer Graphics Forum (Eurographics)*, pp. C97–108.

[12] Burl, M. C. and P. Perona (1996). Recognition of planar object classes. In *In Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn*, pp. 223–230.

[13] Burl, M. C., M. Weber, and P. Perona (1998). A probabilistic approach to object recognition using local photometry and global geometry. pp. 628–641.

[14] Carreira, M., M. Mirmehdi, B. Thomas, and M. Penas (2002). Perceptual primitives from an extended 4D Hough transform. *Image and Vision Computing 20*(13), 969–980.

[15] Chalechale, A., G. Naghdy, and A. Mertins (2005, Jan.). Sketch-based image matching using angular partitioning. *IEEE Transactions on Systems, Man and Cybernetics, Part A 35*(1), 28–41.

[16] Chen, H., Z. Liu, C. Rose, Y. Xu, H.-Y. Shum, and D. Salesin (2004). Example-based composite sketching of human portraits. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 95–153. ACM.

[17] Chung, F. R. K. (1997). *Spectral graph theory*. American Mathematical Society.

[18] Cockshott, T., J. Patterson, and D. England (1992). Modelling the texture of paint. *Computer Graphics Forum 11*(3), 217–226.

[19] Collomosse, J. P. and P. M. Hall (2002). Painterly rendering using image salience. In *Proc. 20th Eurographics UK Conference*, pp. 122–128.

[20] Collomosse, J. P. and P. M. Hall (2003, October). Cubist style rendering from photographs. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 4*(9), 443–453.

[21] Collomosse, J. P. and P. M. Hall (2005, March). Genetic paint: A search for salient paintings. In *proc. EvoMUSART (at EuroGP), Springer Lecture Notes in Computer Science*, Volume 3449, Lausanne, pp. 437–447.

[22] Collomosse, J. P. and P. M. Hall (2006, August). Salience-adaptive painterly rendering using genetic search. *Intl. Journal on Artificial Intelligence Tools (IJAIT) 15*(4), 551–576.

[23] Collomosse, J. P., D. Rowntree, and P. M. Hall (2005, September). Stroke surfaces: Temporally coherent artistic animations from video. *IEEE Transactions on Visualization and Computer Graphics 11*(5), 540–549. ISSN: 1077-2626.

[24] Cour, T., F. Benezit, and J. Shi (2005). Spectral segmentation with multiscale graph decomposition. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, Washington, DC, USA, pp. 1124–1131. IEEE Computer Society.

[25] Crandall, D., P. Felzenszwalb, and D. Huttenlocher (2005, 20–25 June). Spatial priors for part-based recognition using statistical models. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2005*, Volume 1, pp. 10–17.

[26] Csurka, G., C. R. Dance, L. Fan, J. Willamowski, and C. Bray (2004). Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pp. 1–22.

[27] Curtis, C., S. Anderson, J. Seims, K. Fleischer, and D. H. Salesin (1997). Computer-generated watercolor. In *Proc.* 24th *Intl. Conference on Computer Gr aphics and Interactive Techniques (ACM SIGGRAPH)*, pp. 421–430.

[28] Dalal, N. and B. Triggs (2005, 25–25 June). Histograms of oriented gradients for human detection. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2005*, Volume 1, pp. 886–893.

[29] Datta, R., D. Joshi, J. Li, and J. Z. Wang (2008). Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv. 40*(2), 1–60.

[30] DeCarlo, D. and A. Santella (2002). Abstracted painterly renderings using eye-tracking data. In *Proc.* 29th *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, pp. 769–776.

[31] Deussen, O., S. Hiller, C. van Overveld, and T. Strothotte (2000). Floating points: A method for computing stipple drawings. In *Proc. Computer Graphics Forum (Eurographics)*, Volume 19, pp. 41–50.

[32] Dolan, J. and R. Weiss (1989). Perceptual grouping of curved lines. In *Proceedings of a workshop on Image understanding workshop*, San Francisco, CA, USA, pp. 1135–1145. Morgan Kaufmann Publishers Inc.

[33] Elder, J. H. (1999). Are edges incomplete? *Int. J. Comput. Vision 34*(2-3), 97–122.

[34] Elder, J. H. and R. M. Goldberg (2002, 8). Ecological statistics of gestalt laws for the perceptual organization of contours. *J. Vis. 2*(4), 324–353.

[35] Elder, J. H. and S. W. Zucker (1998). Local scale control for edge detection and blur estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 20*(7), 699–716.

[36] Erben, W. (1993). *Miro*. Benedikt Taschen.

[37] Fei-Fei, L., R. Fergus, and P. Perona (2003). A bayesian approach to unsupervised one-shot learning of object categories. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, Washington, DC, USA, pp. 1134. IEEE Computer Society.

[38] Feldman, J. (2003). Perceptual grouping by selection of a logically minimal model. *Int. J. Comput. Vision 55*(1), 5–25.

[39] Felzenszwalb, P. and D. Huttenlocher (2005). Pictorial structures for object recognition. *Int. J. Comput. Vision 1*(61), 55–79.

[40] Fergus, R., P. Perona, and A. Zisserman (2003). Object class recognition by unsupervised scale-invariant learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 2, pp. 264–271.

[41] Ferrari, V., T. Tuytelaars, and L. V. Gool (2006). Object detection by contour segment networks. In *In ECCV*, pp. 14–28.

[42] Fidler, S. and A. Leonardis (2007, 17–22 June). Towards scalable representations of object categories: Learning a hierarchy of parts. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR '07*, pp. 1–8.

[43] Fiore, F. D., F. V. Reeth, J. Patterson, and P. Willis (2007). Highly stylised animation. *The Visual Computer 24*(2), 105–123.

[44] Fischler, M. A. and R. A. Elschlager (1973, Jan.). The representation and matching of pictorial structures. *IEEE Transactions on Computers* (1), 67–92.

[45] Flash, T. and N. Hogan (1985). The coordination of arm movements: An experimentally confirmed mathematical model. *Journal of neuroscience 5*, 1688–1703.

[46] Freeman, W. T. and E. H. Adelson (1991, Sept.). The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*(9), 891–906.

[47] Frome, A., Y. Singer, and J. Malik (2007). Image retrieval and classification using local distance functions. In B. Schölkopf, J. Platt, and T. Hoffman (Eds.), *Advances in Neural Information Processing Systems 19*, pp. 417–424. Cambridge, MA: MIT Press.

[48] Gauch, J. (1999, Jan.). Image segmentation and analysis via multiscale gradient watershed hierarchies. *Image Processing, IEEE Transactions on 8*(1), 69–79.

[49] Gheissari, N. and A. Bab-Hadiashar (2008). A comparative study of model selection criteria for computer vision applications. *Image Vision Comput. 26*(12), 1636–1649.

[50] Gooch, B., G. Coombe, and P. Shirley (2002, June). Artistic vision: Painterly rendering using computer vision techniques. In *Proc.* 2nd *ACM Symposium on Non-photorealistic Animation and Rendering*, pp. 83–90.

[51] Grauman, K. and T. Darrell (2006, 17–22 June). Unsupervised learning of categories from sets of partially matching image features. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 1, pp. 19–25.

[52] Griffin, G., A. Holub, and P. Perona (2007). Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology.

[53] Gutman, I. and B. Zhou (2006). Laplacian energy of a graph. *Linear Algebra and its Applications 44*, 29–37.

[54] Guy, G. and G. Medioni (1996). Inferring global perceptual contours from local features. *Int. J. Comput. Vision 20*(1-2), 113–133.

[55] Haeberli, P. (1990). Paint by numbers: abstract image representations. In *Proc.* 17th *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Volume 4, pp. 207–214.

[56] Haggerty, M. (1991, November). Almost automatic computer painting. *IEEE Computer Graphics and Applications 11*(6), 11–12.

[57] Hall, P. M., J. P. Collomosse, Y. Z. Song, P. Y. Shen, and C. Li (2007). Rtcams: A new perspective on non-photorealistic rendering. *IEEE Trans. on Visualization and Computer Graphics 13*(5), 966–979.

[58] Hall, P. M. and M. J. Owen (2004). Learning to detect low-level features. In *Proceedings Of the BMVC 2004*, pp. 337–356. BMVA.

[59] Haris, K., S. Efstratiadis, N. Maglaveras, and A. Katsaggelos (1998, Dec.). Hybrid image segmentation using watersheds and fast region merging. *Image Processing, IEEE Transactions on 7*(12), 1684–1699.

[60] Harris, C. and M. Stephens (1998). A combined corner and edge detector. In *Proc. 4th Alvey Vision Conference*, Manchester, UK, pp. 189–192.

[61] Hertzmann, A. (1998). Painterly rendering with curved brush strokes of multiple sizes. In *Proc.* 25th *Intl. Conference on Computer Gr aphics and Interactive Techniques (ACM SIGGRAPH)*, pp. 453–460.

[62] Hertzmann, A. (2001, July). Paint by relaxation. In *Proc. Computer Graphics Intl. (CGI)*, pp. 47–54.

[63] Hertzmann, A. (2002, June). Fast paint texture. In *Proc.* 2nd *ACM Symposium on Non-photorealistic Animation and Rendering*, pp. 91–96.

[64] Hertzmann, A., C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin (2001). Image analogies. In *Proc.* 28th *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, pp. 327–340.

[65] Hsu, S. C. and I. H. H. Lee (1994). Drawing and animation using skeletal strokes. In *Proc.* 21st *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Orlando, USA, pp. 109–118.

[66] Huttenlocher, D. P. and S. Ullman (1990). Recognizing solid objects by alignment with an image. *Int. J. Comput. Vision 5*(2), 195–212.

[67] Hyvärinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks 10*(3), 626–634.

[68] Jacobs, C. E., A. Finkelstein, and D. H. Salesin (1995). Fast multiresolution image querying. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, New York, NY, USA, pp. 277–286. ACM.

[69] Kadir, T., A. Zisserman, and M. Brady (2004). An affine invariant salient region detector. In *In Proceedings of the 8th European Conference on Computer Vision*, pp. 228–241.

[70] Kanatani, K. (2004, Oct.). Uncertainty modeling and model selection for geometric inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*(10), 1307–1319.

[71] Kanizsa, G. (1979). *Organization in Vision*. New York: Praeger.

[72] Kirby, M. and L. Sirovich (1990, Jan.). Application of the karhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence 12*(1), 103–108.

[73] Koffka, K. (1935). *Principles of Gestalt Psychology*. New York: Harcourt.

[74] Kohler, W. (1929). *Gestalt Psychology*. New York: Liveright.

[75] Kolesnikov, A. and P. Fränti (2005). Optimal algorithm for convexity measure calculation. In *Int. Conf. Image Processing*, pp. 353–356.

[76] Kumar, M. P., P. H. S. Torr, and A. Zisserman (2004). Extending pictorial structures for object recognition. In *Proceedings of the British Machine Vision Conference*.

[77] Kyprianidis, J. E. and J. Döllner (2008). Image abstraction by structure adaptive filtering. In *Proc. EG UK Theory and Practice of Computer Graphics*, pp. 51–58.

[78] Lamdan, Y., J. T. Schwartz, and H. J. Wolfson (1988, 5–9 June). Object recognition by affine invariant matching. In *Proc. CVPR '88. Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 335–344.

[79] Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998, Nov.). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

[80] Lee, J. (1997). Physically-based modeling of brush painting. *Computer Networks and ISDN Systems 29*, 1571–1756.

[81] Leibe, B. and B. Schiele (2003). Interleaved object categorization and segmentation. In *BMVC*, Volume 2, pp. 759–768.

[82] Leung, T. K., M. C. Burl, and P. Perona (1995, 20–23 June). Finding faces in cluttered scenes using random labeled graph matching. In *Proc. Fifth International Conference on Computer Vision*, pp. 637–644.

[83] Leung, T. K., M. C. Burl, and P. Perona (1998, 23–25 June). Probabilistic affine invariants for recognition. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 678–684.

[84] Litwinowicz, P. (1997). Processing images and video for an impressionist effect. In *Proc.* 24th *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Los Angeles, USA, pp. 407–414. A copy of this paper is included in Appendix C on DVD-ROM.

[85] Loncaric, S. (1998). A survey of shape analysis techniques. *Pattern Recognition 31*(8), 983 – 1001.

[86] Lowe, D. (1987a). Three-dimensional object recognition from single two-dimentional images. *Artificial Intelligence 31*, 355–395.

[87] Lowe, D. G. (1985). *Perceptual Organization and Visual Recognition*. Boston: MA: Kluwer.

[88] Lowe, D. G. (1987b, March). The viewpoint consistency constraint. *International Journal of Computer Vision 1*(1), 57–72.

[89] Lowe, D. G. (1999, 20–27 Sept.). Object recognition from local scale-invariant features. In *Proc. The Seventh IEEE International Conference on Computer Vision*, Volume 2, pp. 1150–1157.

[90] Malpica, N., J. Ortuño, and A. Santos (2003). A multi-channel watershed-based algorithm for supervised texture segmentation. *Pattern Recognition Letters 24*, 1545–1554.

[91] Marr, D. (1981). *VISION: A Computational Investigation into the Human Representation and Processing of Visual Information*. San Francisco: Freeman.

[92] Martin, D., C. Fowlkes, D. Tal, and J. Malik (2001, July). A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, Volume 2, pp. 416–423.

[93] Matas, J., O. Chum, U. Martin, and T. Pajdla (2002, September). Robust wide baseline stereo from maximally stable extremal regions. In P. L. Rosin and D. Marshall (Eds.), *Proceedings of the British Machine Vision Conference*, Volume 1, London, UK, pp. 384–393. BMVA.

[94] McCann, J. and N. S. Pollard (2008). Real-time gradient-domain painting. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, New York, NY, USA, pp. 1–7. ACM.

[95] Meraj, Z., B. Wyvill, T. Isenberg, A. Gooch, and R. Guy (2008). Mimicking hand-drawn pencil lines. In P. Brown, D. W. Cunningham, V. Interrante, and J. McCormack (Eds.), *International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging*, pp. 73–80.

[96] Mikolajczyk, K., B. Leibe, and B. Schiele (2006, 17–22 June). Multiple object class detection with a generative model. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Volume 1, pp. 26–36.

[97] Mikolajczyk, K. and C. Schmid (2001, 7–14 July). Indexing based on scale invariant interest points. In *Proc. Eighth IEEE International Conference on Computer Vision ICCV 2001*, Volume 1, pp. 525–531.

[98] Mikolajczyk, K. and C. Schmid (2002). An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, pp. 128–142. Springer. Copenhagen.

[99] Mikolajczyk, K. and C. Schmid (2005, Oct.). A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 27*(10), 1615–1630.

[100] Mikolajczyk, K., C. Schmid, and A. Zisserman (2004). Human detection based on a probabilistic assembly of robust part detectors. In *European Conference on Computer Vision*, Volume I, pp. 69–81.

[101] Mikolajczyk, K., T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool (2005). A comparison of affine region detectors. *Int. J. Comput. Vision 65*(1-2), 43–72.

[102] Mould, D. (2003). A stained glass image filter. In *EGRW '03: Proceedings of the 14th Eurographics workshop on Rendering*, Aire-la-Ville, Switzerland, Switzerland, pp. 20–25. Eurographics Association.

[103] Mould, D. and K. Grant (2008). Stylized black and white images from photographs. In *NPAR '08: Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 49–58. ACM.

[104] Murase, H. and S. K. Nayar (1995). Visual learning and recognition of 3-d objects from appearance. *Int. J. Comput. Vision 14*(1), 5–24.

[105] Mutch, J. and D. Lowe (2006, 17-22 June). Multiclass object recognition with sparse, localized features. In *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, Volume 1, pp. 11–18.

[106] O'Donovan, P. and D. Mould (2006). Felt-based rendering. In *NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 55–62. ACM.

[107] Ommer, B. and J. M. Buhmann (2003). A compositionality architecture for perceptual feature grouping. *Energy Minimization Methods in Computer Vision and Pattern Recognition*, 275–290.

[108] Opelt, A., M. Fussenegger, A. Pinz, and P. Auer (2004). Weak hypotheses and boosting for generic object detection and recognition. pp. Vol II: 71–84.

[109] Opelt, A., A. Pinz, and A. Zisserman (2006). A boundary-fragment-model for object detection. *European Conference on Computer Vision 2*, 575–588.

[110] Orchard, J. and C. S. Kaplan (2008). Cut-out image mosaics. In *NPAR '08: Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 79–87. ACM.

[111] Orzan, A., A. Bousseau, P. Barla, and J. Thollot (2007). Structure-preserving manipulation of photographs. In *NPAR '07: Proceedings of the 5th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 103–110. ACM.

[112] Papageorgiou, C. P., M. Oren, and T. Poggio (1998, 4–7 Jan.). A general framework for object detection. In *Proc. Sixth International Conference on Computer Vision*, pp. 555–562.

[113] Parent, P. and S. W. Zucker (1989). Trace inference, curvature consistency, and curve detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 11*(8), 823–839.

[114] Perlin, K. and L. Velho (1995). Live paint: Painting with procedural multiscale textures. In *Proc.* 22nd *Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, pp. 153–160.

[115] Pham, B. (1991, January). Expressive brush strokes. *Journal on Graphical Models and Image Processing (CVGIP) 1*(53), 1–6.

[116] Pontil, M., S. Rogai, and A. Verri (1998). Recognizing 3-d objects with linear support vector machines. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume II*, London, UK, pp. 469–483. Springer-Verlag.

[117] Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vettering (1990). *Numerical Recipes in C.* Cambridge University Press.

[118] Pudet, T. (1994). Real time fitting of hand-sketched pressure brushstrokes. In *Proc. Computer Graphics Forum (Eurographics)*, Volume 13, pp. 227–292.

[119] Qu, Y., T.-T. Wong, and P.-A. Heng (2006). Manga colorization. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, New York, NY, USA, pp. 1214–1220. ACM.

[120] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

[121] Ramanan, D., D. A. Forsyth, and A. Zisserman (2005, 20–25 June). Strike a pose: tracking people by finding stylized poses. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2005*, Volume 1, pp. 271–278.

[122] Robels-Kelly, A. and E. R. Hancock (2005). Graph edit distance from spectra seriation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 365–378.

[123] Roberts, L. (1965). Machine perception of 3-d solids. pp. 159–197.

[124] Rosin, P. and C. Mumford (2004, 23–26 Aug.). A symmetric convexity measure. In *Proc. 17th International Conference on Pattern Recognition ICPR 2004*, Volume 4, pp. 11–14.

[125] Rosin, P. L. and G. A. W. West (1995). Curve segmentation and representation by superellipses. *Proc. IEE: Vision, Image, and Signal Processing 142*, 280–288.

[126] Rothwell, C. A., D. A. Forsyth, A. Zisserman, and J. L. Mundy (1993, 11–14 May). Extracting projective structure from single perspective views of 3d point sets. In *Proc. Fourth International Conference on Computer Vision*, pp. 573–582.

[127] Rothwell, C. A., A. Zisserman, D. A. Forsyth, and J. L. Mundy (1995). Planar object recognition using projective shape representation. *Int. J. Comput. Vision 16*(1), 57–99.

[128] Rowley, H. A., S. Baluja, and T. Kanade (1998, Jan.). Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*(1), 23–38.

[129] Salisbury, M., C. Anderson, D. Lischinski, and D. H. Salesin (1996). Scale-dependent reproduction of pen-and-ink illustrations. In *Proc.* 23rd *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, pp. 461–468.

[130] Salisbury, M. P., S. E. Anderson, R. Barzel, and D. H. Salesin (1994). Interactive pen-and-ink illustration. In *Proc.* 21st *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Florida, USA, pp. 101–108.

[131] Salisbury, M. P., M. T. Wong, J. F. Hughes, and D. H. Salesin (1997). Orientable textures for image-based pen-and-ink illustration. In *Proc.* 24th *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Los Angeles, USA, pp. 401–406.

[132] Sarkar, S. and P. Soundararajan (2000). Supervised learning of large perceptual organization: Graph spectral partitioning and learning automata. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 22*(5), 504–525.

[133] Schaffalitzky, F. and A. Zisserman (2001, 7–14 July). Viewpoint invariant texture matching and wide baseline stereo. In *Proc. Eighth IEEE International Conference on Computer Vision ICCV 2001*, Volume 2, pp. 636–643.

[134] Schiele, B. and J. L. Crowley (2000). Recognition without correspondence using multidimensionalreceptive field histograms. *Int. J. Comput. Vision 36*(1), 31–50.

[135] Schmid, C. and R. Mohr (1996, 18–20 June). Combining greyvalue invariants with local constraints for object recognition. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR '96*, pp. 872–877.

[136] Schneiderman, H. and T. Kanade (2000, 13–15 June). A statistical method for 3d object detection applied to faces and cars. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Volume 1, pp. 746–751.

[137] Schroff, F., A. Criminisi, and A. Zisserman (2007, 14–21 Oct.). Harvesting image databases from the web. In *Proc. IEEE 11th International Conference on Computer Vision ICCV 2007*, pp. 1–8.

[138] Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics 6*(2), 461–464.

[139] Setlur, V. and S. Wilkinson (2006). Automatic stained glass rendering. In *Proc. Computer Graphics Intl. (CGI)*, pp. 682–691.

[140] Sharma, G. (2003). *Digital Color Imaging Handbook*. CRC Press.

[141] Shechtman, E. and M. Irani (2007, 17-22 June). Matching local self-similarities across images and videos. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pp. 1–8.

[142] Shi, J. and J. Malik (2000, Aug). Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 22*(8), 888–905.

[143] Shiraishi, M. and Y. Yamaguchi (2000). An algorithm for automatic painterly rendering based on local source image approximation. In *Proc.* 1st *ACM Symposium on Non-photorealistic Animation and Rendering*, pp. 53–58.

[144] Shokoufandeh, A., D. Macrini, S. Dickinson, K. Siddiqi, and S. Zucker (2005, July). Indexing hierarchical structures using graph spectra. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*(7), 1125–1140.

[145] Shotton, J., A. Blake, and R. Cipolla (2005). Contour-based learning for object detection. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Washington, DC, USA, pp. 503–510. IEEE Computer Society.

[146] Shotton, J., A. Blake, and R. Cipolla (2008, July). Multiscale categorical object recognition using contour fragments. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*(7), 1270–1281.

[147] Shugrina, M., M. Betke, and J. Collomosse (2006). Empathic painting: interactive stylization through observed emotional state. In *NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 87–96. ACM.

[148] Sinclair, D. (1999). Voronoi seeded colour image segmentation. Technical Report TR99-04, AT&T Laboratories Cambridge.

[149] Sivic, J. (2006). *Efficient visual search of images and videos.* Ph. D. thesis, Robotics Research Group, Department of Engineering Science, University of Oxford.

[150] Sivic, J., B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman (2005a). Discovering object categories in image collections. In *Proceedings of the International Conference on Computer Vision.*

[151] Sivic, J., B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman (2005b, 17–21 Oct.). Discovering objects and their location in images. In *Proc. Tenth IEEE International Conference on Computer Vision ICCV 2005*, Volume 1, pp. 370–377.

[152] Small, D. (1991, February). Simulating watercolor by modeling diffusion, pigment and paper fibres. In *Proc. SPIE*, pp. 70–76.

[153] Smeulders, A. W. M., M. Worring, S. Santini, A. Gupta, and R. Jain (2000). Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell. 22*(12), 1349–1380.

[154] Sousa, M. C. and J. W. Buchanan (1999a). Computer-generated graphite pencil rendering of 3D polygonal models. In *Proc. Computer Graphics Forum (Eurographics)*, Volume 3, pp. 195–208.

[155] Sousa, M. C. and J. W. Buchanan (1999b, June). Observational models of blenders and erasers in computer-generated pencil rendering. In *Proc. Graphics Interface*, pp. 157–166.

[156] Sousa, M. C. and J. W. Buchanan (2000, March). Observational models of graphite pencil materials. *Computer Graphics Forum 1*(19), 27–49.

[157] Strassmann, S. (1986). Hairy brushes. In *Proc.* 13$^{\text{th}}$ *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Volume 20, pp. 225–232.

[158] Sziranyi, T. and Z. Tath (2000). Random paintbrush transformation. In *Proc.* 15$^{\text{th}}$ *Intl. Conference on Pattern Recognition (ICPR)*, Volume 3, Barcelona, pp. 155–158.

[159] Takagi, S., M. Nakajima, and I. Fujishiro (1999, October). Volumetric modeling of colored pencil drawing. In *Proc.* 7$^{\text{th}}$ *Pacific Conference on Computer Graphics and Applications*, Seoul, Korea, pp. 250.

[160] Todorovic, S. and N. Ahuja (2008). Region-based hierarchical image matching. *Int. J. Comput. Vision 78*(1), 47–66.

[161] Torr, P. H. S. (1997, 17–19 June). An assessment of information criteria for motion model selection. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 47–52.

[162] Torralba, A., K. P. Murphy, and W. T. Freeman (2004, 27 June–2 July). Sharing features: efficient boosting procedures for multiclass object detection. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2004*, Volume 2, pp. II–762–II–769.

[163] Town, C. and D. Sinclair (2004). Language-based querying of image collections on the basis of an extensible ontology. *International Journal of Image and Vision Computing 22(3)*, 251–267.

[164] Treavett, S. and M. Chen (1997, March). Statistical techniques for the automated synthesis of non-photorealistic images. In *Proc.* 15th *Eurographics UK Conference*, pp. 201–210.

[165] Turk, M. A. and A. P. Pentland (1991, 3–6 June). Face recognition using eigenfaces. In *Proc. CVPR '91. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 586–591.

[166] Tuytelaars, T. and L. V. Gool (2000). Wide baseline stereo matching based on local, affinely invariant regions. In *In Proc. BMVC*, pp. 412–425.

[167] Vincent, L. and P. Soille (1991). Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*(6), 583–598.

[168] Viola, P. and M. Jones (2001). Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR 2001*, Volume 1, pp. I–511–I–518.

[169] Voss, K. and H. Suesse (1997, Jan.). Invariant fitting of planar objects by primitives. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*(1), 80–84.

[170] Wang, J., Y. Xu, H.-Y. Shum, and M. F. Cohen (2004). Video tooning. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, New York, NY, USA, pp. 574–583. ACM.

[171] Weber, M., W. Einhauser, M. Welling, and P. Perona (2000, 28–30 March). Viewpoint-invariant learning and detection of human heads. In *Proc. Fourth IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 20–27.

[172] Weber, M., M. Welling, and P. Perona (2000a, 13–15 June). Towards automatic discovery of object categories. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Volume 2, pp. 101–108.

[173] Weber, M., M. Welling, and P. Perona (2000b). Unsupervised learning of models for recognition. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part I*, London, UK, pp. 18–32. Springer-Verlag.

[174] Wen, F., Q. Luan, L. Liang, Y.-Q. Xu, and H.-Y. Shum (2006). Color sketch generation. In *NPAR '06: Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 47–54. ACM.

[175] Wertheimer, M. (1923). *Laws of Organization in Perceptual Forms*, Volume 4. Psycologische Forschung. Translation published in Ellis, W. (1938).

[176] Whitted, T. (1983, July). Anti-aliased line drawing using brush extrusion. In *Proc.* 10th *Intl. Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH)*, Detroit, Michigan, pp. 151–156.

[177] Winnemöller, H., S. C. Olsen, and B. Gooch (2006). Real-time video abstraction. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, New York, NY, USA, pp. 1221–1226. ACM.

[178] Witkin, A. and J. Tenenbaum (1983). On the role of structure in vision. In B. H. J. Beck and A. Rosenfeld (Eds.), *Human and Machine Vision*, pp. 481–543. New York: Academic.

[179] Witkin, A. P. (1987). Scale-space filtering. pp. 329–332.

[180] Wolfson, H. J. and I. Rigoutsos (1997, Oct.–Dec.). Geometric hashing: an overview. *IEEE Computational Science & Engineering 4*(4), 10–21.

[181] Xu, J. and C. S. Kaplan (2008). Artistic thresholding. In *NPAR '08: Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, New York, NY, USA, pp. 39–47. ACM.

[182] Xu, J., C. S. Kaplan, and X. Mi (2007). Computer-generated papercutting. In *PG '07: Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, Washington, DC, USA, pp. 343–350. IEEE Computer Society.

[183] Xu, S., F. C. M. Lau, F. Tang, and Y. Pan (2003, September). Advanced design for a realistic virtual brush. In *Proc. Computer Graphics Forum (Eurographics)*, Volume 3, Grenada, Spain, pp. 533–542.

[184] Xu, S., F. Tang, F. C. M. Lau, and Y. Pan (2002, September). A solid model based virtual hairy brush. In *Proc. Computer Graphics Forum (Eurographics)*, Volume 21, pp. 299–308.

[185] Zhang, J., M. Marszalek, S. Lazebnik, and C. Schmid (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. Comput. Vision 73*(2), 213–238.

[186] Zhu, P. and R. Wilson (2005). Stability of the eigenvalues of graphs. In *The 11th International Conference on Computer Analysis of Images and Patterns*, pp. 371–377.